




ADVERTIMENT. L'accés als continguts d'aquesta tesi queda condicionat a l'acceptació de les condicions d'ús establertes per la següent llicència Creative Commons:  <https://creativecommons.org/licenses/?lang=ca>

ADVERTENCIA. El acceso a los contenidos de esta tesis queda condicionado a la aceptación de las condiciones de uso establecidas por la siguiente licencia Creative Commons:  <https://creativecommons.org/licenses/?lang=es>

WARNING. The access to the contents of this doctoral thesis it is limited to the acceptance of the use conditions set by the following Creative Commons license:  <https://creativecommons.org/licenses/?lang=en>



**Universitat Autònoma
de Barcelona**

**Improving Mapping and Localization using Deep
Learning**

A dissertation submitted by **Mohammad Altillawi**
at Universitat Autònoma de Barcelona to fulfil the
degree of **Doctor of Philosophy**.

Bellaterra, July 11, 2024

Tutors	Prof. Joan Serrat Universitat Autònoma de Barcelona Dept. Ciències de la Computació Centre de Visió per Computador
Directors	Dr. Shile Li Huawei-Munich Research Center Munich, Germany
Thesis committee	President institution city, country Secretary institution city, country Vocal institution city, country



This document was typeset by the author using \LaTeX 2 ϵ .

The research described in this book was carried out at the Computer Vision Center, Universitat Autònoma de Barcelona.

This work is licensed under Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) © 2020 by **Mohammad Altilawi**. You are free to copy and redistribute the material in any medium or format as long as you attribute its author. If you alter, transform or build upon this work, you may distribute the resulting work only under the same, similar or compatible license.

ISBN 978-84-124793-8-6

Printed by IMPRIMA

Dedication...

Acknowledgments

Abstract

Localizing a camera from a single image in a previously visited area enables applications in many fields, such as robotics, autonomous driving, and augmented/virtual reality. With the recent advancements in computational resources and the rapid emergence of accompanying open-source machine learning frameworks, deep learning has swiftly gained traction within both research communities and industries. As a result, recent works opted for data-driven solutions for the problems at hand. In this dissertation, we propose solutions to utilize deep learning to improve camera re-localization from a single image. After the introductory chapter, we define and address the limitations of current approaches and provide data-driven solutions in the following chapters.

In chapter two, we propose a method that learns a geometric prior to inform about reliable regions in a given scene. It selects reliable pixel features from the input single image and estimates their corresponding 3D Coordinates directly for pose estimation. In essence, the proposed method does not consider the whole image as useful for localization. It avoids selecting keypoints (thus the corresponding 3D points) from non-discriminative image regions such as the sky and trees, dynamic objects such as cars and pedestrians, and occlusions. By bypassing these outlier sources, the proposed method selects a controllable number of correspondences, enhancing localization efficiency and accuracy.

In chapter three, we propose to leverage a novel network of spatial and temporal geometric constraints in the form of relative camera poses that are obtained from adjacent and distant cameras to guide the training of a deep network for localization. In this context, the deep network acts as the map of the scene. By employing our constraints, we guide the network to encode these geometric constraints in its weights. By encompassing these encoded constraints, the localization pipeline obtains better accuracy at inference.

In chapter four, we propose a novel pipeline to utilize the minimal available labels (i.e., poses) for a data-driven pose estimation pipeline to obtain the geometry of the scene. The proposed framework uses a differentiable rigid alignment module to pass gradients into the deep network to adjust the learned geometry. It learns two 3D geometric representations (X, Y, Z coordinates) of the scene, one in camera coordinate frame and the other in global coordinate frame. Given a single image at inference, the proposed pipeline triggers the deep network to obtain the two geometric representations of the observed scene and aligns them to estimate the pose of the camera. As a

result, the proposed method learns and incorporates geometric constraints for a more accurate pose estimation.

In chapter five, we propose to explore the power of generative models for data generation for data-driven localization. We thus contribute with a novel data-centric method to generate correspondences across different viewing and illumination conditions to enhance the robustness of localization towards long-term changes (daytime, weather, season). The proposed method represents the scene with a number of implicit representations (based on NeRFs), each corresponds to different illumination condition. Consequently, it utilizes the underlying geometry based on these representations to generate accurate correspondences across the different illumination variations. Using these correspondences enhances localization across long-term changes. Besides, we built an evaluation benchmark to assess and evaluate the performances of feature extraction and description networks towards localization across long-term illumination changes. Our work serves as a substantial stride toward robust long-term localization.

In chapter six, we propose a point-based generative model to synthesize novel views. The proposed work points out and solves a mismatch issue between geometry (point cloud) and appearance (images), which generates degraded renderings. The proposed method employs a connectivity graph between appearance and geometry. In contrast to using the whole point cloud of a scene, it retrieves points from a large point cloud that are observed from the current camera perspective and uses them for rendering. This makes the rendering pipeline faster and more scalable. We emphasize as well the power of this connectivity graph to the recent 3D Gaussian splatting scene representation. Our proposal employs image reconstruction with generative adversarial training for enhanced rendering quality. Such a pipeline can be used to generate novel views to augment/create more labeled data for pose estimation.

Keywords – Camera Localization, Keypoint detection, implicit geometric estimation, point cloud alignment, point-based rendering, long-term localization, deep learning, AI generated content.

Resum

Localitzar una càmera des d'una única imatge en una àrea visitada prèviament te aplicació en en molts camps, com ara la robòtica, la conducció autònoma i la realitat augmentada/virtual. Amb els recents avenços en recursos computacionals i l'aparició d'entorns d'aprenentatge de codi obert, l'aprenentatge profund ha guanyat ràpidament impuls tant en comunitats de recerca com en indústries. Com a resultat, els treballs recents han optat per solucions basades en dades pels problemes que tenim. En aquesta tesi, proposem solucions per utilitzar l'aprenentatge profund per millorar la re-localització de la càmera des d'una sola imatge. Després del capítol introductori, definim i afrontem les limitacions dels enfocaments actuals i proporcionem solucions impulsades per Dades en els següents capítols.

En el segon capítol proposem un mètode que aprèn un prior geomètric per a detectar regions fiables en una escena determinada. Aquest mètode selecciona característiques de píxels fiables de la imatge única d'entrada i estima les seves coordenades 3D corresponents directament per a l'estimació de la posició i orientació de la càmera. En essència, el mètode proposat no considera que tota la imatge sigui útil per a la localització. Evita seleccionar punts clau (per tant, els punts 3D corresponents) de regions d'imatge no discriminats com el cel i els arbres, objectes dinàmics com cotxes i vianants, i les oclusions. En evitar aquestes fonts atípiques, el mètode proposat selecciona un nombre controlable de correspondències, millorant l'eficiència de la localització i la seva precisió.

En el tercer capítol proposem aprofitar una nova xarxa de restriccions geomètriques espacials i temporals en forma de poses de càmera relativa que s'obtenen de càmeres adjacents i distants per tal de guiar la formació d'una xarxa profunda per a la localització. En aquest context, la xarxa profunda actua com el mapa de l'escena. Utilitzant les nostres restriccions, guiem la xarxa per codificar aquestes restriccions geomètriques en els seus pesos. En incloure aquestes restriccions codificades, la procés de localització obté una millor precisió a la inferència.

En el quart capítol proposem una nova estratègia per a utilitzar les etiquetes mínimes disponibles (és a dir, la pose) per a un mètode d'estimació de posició impulsada per dades per obtenir la geometria de l'escena. El marc de treball proposat utilitza un mòdul d'alineació rígid diferenciable per passar gradients a la xarxa profunda per tal d'ajustar la geometria apresada. Aprèn dues representacions geomètriques en 3D (*coordenades* X, Y, Z) de l'escena, una en el marc de coordenades de la càmera i l'altra en el marc de coordenades global. Donada una única imatge a la inferència, el mètode

proposat activa la xarxa profunda per a obtenir les dues representacions geomètriques de l'escena observada i les alinea per estimar la posició de la càmera. Com a resultat, el mètode proposat aprèn i incorpora restriccions geomètriques per a una estimació de posició més precisa.

En el cinquè capítol proposem explorar la potència dels models generatius per a la generació de dades per a la localització basada en dades. Contribuïm així amb un nou mètode de centrat en dades per a generar correspondències en diferents condicions de visualització i il·luminació per potenciar la robustesa de la localització cap a canvis a llarg termini (dia-nit, meteorològic, estacional). El mètode proposat representa l'escena amb un nombre de representacions implícites (basades en NeRFs), cadascuna correspon a una condició d'il·luminació diferent. En conseqüència, utilitza la geometria subjacent basada en aquestes representacions per generar correspondències precises entre les diferents variacions d'il·luminació. En utilitzar aquestes correspondències millora la localització davant dels canvis a llarg termini. A més, vam construir un punt de referència per avaluar i avaluar els rendiments de les xarxes d'extracció i descripció de característiques cap a la localització en els canvis d'il·luminació a llarg termini. El nostre treball serveix com un pas substancial cap a una localització robusta a llarg termini.

En el sisè capítol proposem un model generatiu basat en punts per sintetitzar punts de vista novells. El treball proposat apunta i resol un problema desajustament entre la geometria (núvol de punt) i l'aparença (imatges), que genera representacions degradades. El mètode proposat utilitza un gràfic de connectivitat entre aparença i geometria. En contrast amb l'ús de tot el núvol de punts d'una escena, recupera punts d'un gran núvol de punts que s'observen des de la perspectiva de la càmera actual i els utilitza per a la representació. Això fa que el mètode de representació sigui més ràpid i escalable. També ressaltam la potència d'aquesta gràfica de connectivitat a la recent representació d'escenes en 3D gaussiana. La nostra proposta utilitza la reconstrucció d'imatges amb un entrenament adversari generatiu per millorar la qualitat de representació. Aquest mètode es pot utilitzar per generar vistes noves per augmentar/crear més dades etiquetades per a l'estimació de posició.

Paraules clau – Localització de la càmera, detecció de punts clau, estimació geomètrica implícita, alineació de núvols de punts, representació basada en punts, localització a llarg termini, aprenentatge profund, contingut generat per IA.

Resumen

Localizar una cámara a partir de una sola imagen en una zona previamente visitada permite aplicaciones en muchos campos, como la robótica, la conducción autónoma y la realidad aumentada/virtual. Con los recientes avances en recursos computacionales y la rápida aparición de marcos de aprendizaje automático de código abierto, el aprendizaje profundo ha ganado adeptos tanto en las comunidades de investigación como en la industria. Como resultado, los últimos trabajos han optado por soluciones basadas en datos para los problemas planteados. En esta tesis, proponemos soluciones para utilizar el aprendizaje profundo para mejorar la relocalización de la cámara a partir de una sola imagen. Después del capítulo introductorio, definimos y abordamos las limitaciones de los enfoques actuales y proporcionamos soluciones basadas en datos en los siguientes capítulos

En el segundo capítulo proponemos un método que aprende un prior geométrico para detectar regiones fiables en una escena dada. Selecciona características de píxeles fiables de la imagen única de entrada y estima sus correspondientes coordenadas 3D directamente para la estimación de la pose. En esencia, el método propuesto no considera toda la imagen útil para la localización. Evita seleccionar puntos clave (y por tanto los correspondientes puntos 3D) de regiones de la imagen no informativas, como el cielo y los árboles, objetos dinámicos como coches y peatones, y oclusiones. Al evitar estas fuentes atípicas, el método propuesto selecciona un número controlable de correspondencias, mejorando la eficiencia y la precisión de la localización.

En el capítulo tres proponemos aprovechar una novedosa red de restricciones geométricas espaciales y temporales en forma de poses relativas de cámara que se obtienen de cámaras adyacentes y distantes para guiar el entrenamiento de una red profunda para la localización. En este contexto, la red profunda actúa como el mapa de la escena. Al emplear nuestras restricciones, guiamos a la red para que codifique estas restricciones geométricas en sus pesos. Al englobar estas restricciones codificadas, el proceso de localización obtiene una mayor precisión en la inferencia.

En el capítulo cuarto proponemos un nuevo proceso para utilizar las etiquetas mínimas disponibles (es decir, poses) para un proceso de estimación de poses basado en datos para obtener la geometría de la escena. El marco propuesto utiliza un módulo de alineación rígida diferenciable para pasar gradientes a la red profunda y ajustar la geometría aprendida. Aprende dos representaciones geométricas 3D (*coordenadas X, Y, Z*) de la escena, una en el marco de coordenadas de la cámara y la otra en el marco de coordenadas global. Dada una única imagen en la inferencia, el conducto propuesto

activa la red profunda para obtener las dos representaciones geométricas de la escena observada y las alinea para estimar la pose de la cámara. Como resultado, el método propuesto aprende e incorpora restricciones geométricas para una estimación de pose más precisa.

En el capítulo quinto proponemos explorar el poder de los modelos generativos para la generación de datos para la localización basada en datos. Así, contribuimos con un novedoso método centrado en los datos para generar correspondencias a través de diferentes condiciones de visión e iluminación para mejorar la robustez de la localización frente a cambios a largo plazo (diurno-nocturno, meteorológicos, estacionales). El método propuesto representa la escena con una serie de representaciones implícitas (basadas en NeRFs), cada una de las cuales corresponde a diferentes condiciones de iluminación. En consecuencia, utiliza la geometría subyacente basada en estas representaciones para generar correspondencias precisas a través de las diferentes variaciones de iluminación. El uso de estas correspondencias mejora la localización a través de cambios a largo plazo. Además, hemos creado un sistema de evaluación comparativa para evaluar el rendimiento de las redes de extracción de características y de descripción en la localización a través de cambios de iluminación a largo plazo. Nuestro trabajo supone un avance sustancial hacia una localización robusta a largo plazo. v

En el capítulo sexto proponemos un modelo generativo basado en puntos para sintetizar nuevas vistas. El trabajo propuesto señala y resuelve un problema de desajuste entre la geometría (nube de puntos) y la apariencia (imágenes), que genera visualizaciones degradadas. El método propuesto emplea un gráfico de conectividad entre la apariencia y la geometría. En lugar de utilizar toda la nube de puntos de una escena, recupera puntos de una gran nube de puntos que se observan desde la perspectiva de la cámara actual y los utiliza para el renderizado. Esto hace que el proceso de renderizado sea más rápido y escalable. Destacamos también la potencia de este gráfico de conectividad para la reciente representación de escenas 3D Gaussian *splatting*. Nuestra propuesta emplea la reconstrucción de imágenes con entrenamiento generativo adversario para mejorar la calidad del renderizado. Este proceso puede utilizarse para generar nuevas vistas que aumenten/creen más datos etiquetados para la estimación de la pose.

Palabras clave – Localización de cámaras, detección de puntos clave, estimación geométrica implícita, alineación de nubes de puntos, renderizado basado en puntos, localización a largo plazo, aprendizaje profundo, contenido generado por IA.

Contents

1	Introduction	1
1.1	Motivation and Preliminaries	1
1.1.1	Motivation	1
1.1.2	Formulation	2
1.1.3	Data Preparation	3
1.1.4	Related Work	3
1.2	Outline, Research Questions and Contributions	4
1.2.1	Chapter 2: Learning Geometric Prior About Reliable Pixels for Absolute Localization	4
1.2.2	Chapter 3: Spatio-Temporal Constraints Across Distant Cameras for Absolute Localization	5
1.2.3	Chapter 4: Learning Scene Geometry without Direct Labels for Absolute Localization	5
1.2.4	Chapter 5: Cross-Domain Correspondences for Long-Term Localization	6
1.2.5	Chapter 6: Point-based Large-Scale Novel View Synthesis	6
1.3	List of Publications	7
1.4	List of Patents	7
2	PixSelect: Less but Reliable Pixels for Accurate and Efficient Localization[†]	9
2.1	Introduction	10
2.2	Related Work	11
2.3	Method	13
2.3.1	Localization Pipeline	14
2.4	Experiments and Evaluation	16
2.4.1	Datasets	16
2.4.2	Architecture and Setup	16
2.4.3	Baselines	17
2.4.4	Comparison of Localization Errors Against State of Art Methods	17
2.4.5	PixSelect as Outlier Filter	19
2.4.6	Importance of Selecting Reliable Keypoints	19
2.4.7	Observations	21
2.5	Conclusions	21

2.6	Further Results	23
3	Global Localization: Utilizing Relative Spatio-Temporal Geometric Constraints from Adjacent and Distant Cameras[†]	25
3.1	Introduction	26
3.2	Related Works	28
3.3	Method	29
3.3.1	Overview	29
3.3.2	3D Points in Global Coordinate Frame	31
3.3.3	3D Points in Camera Coordinate Frame	31
3.3.4	Pose Estimation	32
3.3.5	Distant and Adjacent Spatio-Temporal Constraints:	32
3.3.6	Weights	34
3.4	Experiments	35
3.4.1	Datasets	35
3.4.2	Architecture and Setup	35
3.4.3	Learning From Little or Sparse Ground-truth	36
3.4.4	Effectiveness of the Distant Spatio-Temporal Constraints	38
3.4.5	Comparison Against State-of-the-Art Methods	40
3.5	Conclusions	41
4	Implicit Learning of Scene Geometry from Poses for Global Localization[†]	43
4.1	Introduction	44
4.2	Related Work	46
4.3	Method	47
4.3.1	Overview	47
4.4	Results	50
4.4.1	Datasets	50
4.4.2	Setup	51
4.4.3	Results: Comparison to Previous Methods	51
4.4.4	Results: Ablation Study	53
4.4.5	Results: Outliers Filtering	55
4.4.6	Results: Finetuning With Position Labels Only	57
4.4.7	Results: Run-time	58
4.5	Conclusions	59
5	Long-Term Invariant Local Features via Implicit Cross-Domain Correspondences[†]	61
5.1	Introduction	62
5.2	Related Work	64
5.3	Optimizing the 4Seasons Dataset to Benchmark Cross-Domain Visual Localization	66
5.3.1	Original 4Seasons Dataset Details	66
5.3.2	Refining the 4Seasons Dataset	67
5.3.3	Refined Map Quality Analysis	69
5.4	Benchmarking Localization Robustness Across Visual Domains	71

5.4.1	Benchmarking Relative Pose Estimation	71
5.4.2	Benchmarking Absolute Pose Estimation	73
5.5	Network Supervision for Long-Term Invariant Descriptors	75
5.5.1	Correspondence Generation Overview	75
5.5.2	iCDC: Implicit Cross-Domain Correspondences	76
5.6	Experimental Setup Details	79
5.6.1	Train-test-split	79
5.6.2	Constructed Maps	79
5.6.3	iCDC Setup	80
5.6.4	Extractor Training Setup	80
5.7	Results	81
5.7.1	Overview of Experiments	81
5.7.2	Qualitative Correspondence Analysis	82
5.7.3	Relative Pose Estimation on Our Benchmark	83
5.7.4	Absolute Pose Estimation on Our Benchmark	85
5.7.5	Evaluation of Our Correspondences on the Visual Localization Benchmark	86
5.8	Additional Insights	89
5.8.1	Training Learned Matchers Using iCDC	89
5.8.2	Additional Training Insights	94
5.9	Conclusions	96
6	Large-Scale Novel View Synthesis with Enhanced Neural Point-Based Graphics[†]	99
6.1	Introduction	100
6.2	Related Work	102
6.3	Method	104
6.4	Experiments and Evaluation	109
6.4.1	Datasets	109
6.4.2	Baselines Methods	110
6.4.3	Training Details	110
6.4.4	Comparison to Previous Neural Point Based Rendering Methods	111
6.4.5	Usability and scalability of our method	112
6.5	Conclusions	116
7	Conclusions and Future Directions	117
7.1	Conclusions	117
7.2	Future Work	119
	List of Contributions	123
7.3	List of Publications	123
7.4	List of Patents	123
	Bibliography	125

List of Tables

2.1	Comparison against State-of-the-art visual localization methods on the Cambridge Landmarks dataset[70]. We report the median translation and rotation errors.	18
2.2	Comparison against State-of-the-art visual localization methods on the Cambridge Landmarks dataset[70]. We report the median translation and rotation errors.	18
2.3	Median position errors on Cambridge Landmarks of our work against Neural guided RANSAC [17]. The results show that the correspondences obtained by our work incur a low number of outliers. It acts naturally as an outlier filter.	20
2.4	Localization median errors for two sets of correspondences. Set 1: The 200 correspondences of high confidence. Set 2: 200 correspondences with lower confidence. Refer to Fig. 2.3 for visual feedback.	20
3.1	Results of the experiment of Sec. 3.4.3. The second row lists the average number of available ground-truth 3D coordinates per image and the corresponding % of the total number of possible ground-truth (GT) points (6420) for output resolution of 60×107	37
3.2	The table shows the results of the experiment of Sec. 3.4.4 on the first six scenes of the 12Scenes dataset [147]. The results on the remaining scenes are listed in Tab. 3.3. Errors are reported as median translation error (meters)/median rotation error (degrees). The second row reports, for each scene, the average percentage of available ground-truth (GT) 3D coordinates of the total number of possible ground-truth points per image. The best results are marked in bold.	39
3.3	The table shows the results of the experiment of Sec. 3.4.4 on the second half of scenes of the 12Scenes dataset [147]. The results on the first six scenes are listed in Tab. 3.2. The same caption of Tab. 3.2 applies here. . .	39
3.4	Comparison against State-of-the-art localization methods on the 7scenes [132]. Median translation (m) and rotation ($^{\circ}$) errors (lower is better) with the improvements relative to the second best-reported quantity (underlined) are reported.	40

3.5	Localization errors on the Cambridge Landmarks [70]. Median translation (m) and rotation (°) errors (lower is better) with the improvements relative to the second best-reported quantity (underlined) are reported.	41
4.1	Comparison against state-of-the-art localization methods on Cambridge Landmarks [70]. The first and second best results are marked in bold and <u>underline</u> , respectively. ' _ ' denotes unavailable results.	52
4.2	Comparison against state-of-the-art localization methods on 7Scenes dataset [132]. The first and second best results are marked in bold and <u>underline</u> , respectively. ' _ ' denotes unavailable results.	53
4.3	Ablation results of section 4.4.4 on Cambridge Landmarks [70], 7Scenes [132], and 12Scenes [147] datasets.	54
4.4	Effect of different filtering methods (section 4.4.5). The best results are marked in bold.	56
4.5	Results of the experiment of section 4.4.6 on Cambridge Landmarks [70]. Median errors (meters/degrees) are reported. Improvements as a result of finetuning by $L_{position}$ over training on 1/3 of samples are marked by underlines.	57
4.6	Results of the experiment of section 4.4.6 on 7Scenes dataset [132]. Median errors (meters/degrees) are reported. Improvements as a result of finetuning by $L_{position}$ over training on 1/3 of samples are marked by underlines.	58
4.7	Run-time analysis (section 4.4.7). Data Processing: time needed to run the network and obtain the 3D clouds. Pose Computation: time needed to obtain pose through rigid alignment. FPS: frames per second.	58
5.1	Dataset overview: trajectory labels are used in this work to refer to the corresponding trajectories. The domain is described for each trajectory. For each scene, the average number of frames with RTK-GNSS measurements per trajectory is provided, as well as the average of the total number of available frames.	66
5.2	Relative pose errors evaluated on 4Seasons poses vs. our refined poses across different illumination conditions. Pose errors are presented both as median errors and AUC accuracies. Consistent higher performances when evaluated against our refined poses indicates higher ground-truth pose quality.	71
5.3	Relative pose errors evaluated on 4Seasons poses vs. our refined poses under the same illumination conditions. Pose errors are presented both as median errors and AUC accuracies. Consistent higher performances when evaluated against our refined poses indicates higher ground-truth pose quality.	71

5.4	Median relative pose errors on the 4Seasons dataset [156]. For each scene, aggregated median errors are presented and evaluated against our ground truth poses. In the case of cross-domain, the error under reference trajectory is the mean of all median pose errors between the reference and all query trajectories.	72
5.5	Cross-domain absolute pose accuracies using ground-truth poses for image retrieval on the 4Seasons dataset [156]. For each scene, the percentages of accurately localized query images under thresholds (0.25m, 2°), (0.5m, 5°), and (5m, 10°) are evaluated and reported against our ground truth poses.	74
5.6	Cross-domain absolute pose accuracies using NetVLAD [6] for image retrieval on the 4Seasons dataset [156]. For each scene, the percentages of accurately localized query images under thresholds (0.25m, 2°), (0.5m, 5°), and (5m, 10°) are evaluated and reported against our ground truth poses.	74
5.7	Cross-domain absolute pose accuracies on the 4Seasons dataset [156]. For our test scene, the percentages of accurately localized query images under thresholds (0.25m, 2°), (0.5m, 5°), and (5m, 10°) are evaluated and reported against our ground truth poses. Results using default weights, our trained R2D2 networks, and our trained SiLK networks are grouped separately.	86
5.8	Benchmarking long-term visual localization on CMU Seasons Extended dataset. We evaluate the performance of our extractors on the long-term localization benchmark [124] and compare them to the performances of SuperPoint, R2D2, and SiLK with default weights.	87
5.9	Benchmarking long-term visual localization on RobotCar seasons dataset. We evaluate the performance of our extractors on the long-term localization benchmark [124] and compare them to the performances of SuperPoint, R2D2, and SiLK with default weights.	87
5.10	Benchmarking long-term visual localization on Aachen Day-Night dataset. We evaluate the performance of our extractors on the long-term localization benchmark [124] and compare them to the performances of SuperPoint, R2D2, and SiLK with default weights. Aachen Day-Night presents the greatest challenge due to the large domain variation between handheld and car-mounted camera setups.	88
5.11	Benchmarking long-term visual vocalization on CMU Seasons Extended dataset [8]. Performances of network matcher pairs are compared. NN means Euclidean distance-based nearest neighbor matching. LGlue refers to LightGlue [86]. Nov. means the network was trained using rendered novel views of our NeRFs.	91
5.12	Benchmarking long-term visual vocalization on RobotCar-Seasons dataset [90]. Performances of network matcher pairs are compared. NN means Euclidean distance-based nearest neighbor matching. LGlue refers to LightGlue [86]. Nov. means the network was trained using rendered novel views of our NeRFs.	92

5.13	Benchmarking long-term visual vocalization on Aachen Day-Night [125]. Performances of network matcher pairs are compared. NN means Euclidean distance-based nearest neighbor matching. LGlue refers to Light-Glue [86]. Nov. means the network was trained using rendered novel views of our NeRFs.	92
5.14	Detailed CMU Seasons [8] benchmark results	93
5.15	Detailed RobotCar-Seasons [90] benchmark results	93
6.1	Details of the sequences from KITTI360 that are used in our experiments .	110
6.2	Comparison of our method against the state-of-the-art neural point-based rendering methods on sequences from the KITTI360 dataset.	111
6.3	Utilizing our connectivity relationship graph for 3D Gaussian Splatting [71] scene fitting.	113
6.4	Comparison of our neural point-based method and 3D Gaussian splatting (3DGS) for novel view synthesis on a sequence from the KITTI360 dataset. For 3DGS, we use our connectivity setup.	115
6.5	Run-time improvements from using our connectivity graph on 3DGS. . . .	115

List of Figures

2.1	Not all image regions are equally important for visual localization. Our method, PixSelect, selects 2D keypoints from discriminatory image regions that are useful for localization. The left side images show the valid keypoints obtained by PixSelect for the sake of localization. The right side images are the corresponding predicted heatmaps, with good image regions highlighted in yellowish-green (best seen in colors). Our method can emphasize reliable regions for localization and avoid other nonrelevant regions such as the trees, the cloudy sky, and repetitive plane structures like walls, in addition to pedestrians and cars. We draw your attention, in particular, to the bottom image and the corresponding heatmap, where it can be observed that our method is able to distinguish the trees from the structure (building). While emphasizing the background building, it does not select keypoints from the tree branches. It also ignores the reflective rounded windows.	12
2.2	A sketched representation of our pipeline: The processing flow is left to right. For a given input image, a deep network produces two outputs. The first is a set of reliable 2D keypoints. Here, we already show the valid 2D keypoints that are overlayed on the image. The second is a set of 3D coordinates relative to a global coordinate system. Here, we show an illustrative example. Correspondences are obtained directly by selecting the 3D coordinates corresponding to the valid 2D keypoints. The set of selected 2D-3D correspondences is then used to obtain a pose using Perspective-n-Point [47] in a RANSAC scheme [43]. The most right sketch shows the pose computation step of the camera relative to a scene where the green lines describe the 2D-3D correspondences. For a description of the network architecture, refer to section 4.4.2.	14
2.3	Importance of selected Keypoints: The upper images show keypoints that are predicted with lower scores (set 2 of Tab. 2.4). The lower images show keypoints that are predicted with higher scores (set 1 of Tab. 2.4). Green color denotes inliers, the set of keypoints that resulted in the pose hypothesis from RANSAC. Outliers are denoted by red. Left: King’s College, right: Old Hospital.	21

- 2.4 Keypoints selected from the GreatCourt landmark. The grass, the building's roofs, and the sky are avoided for being non-discriminatory and thus irrelevant for localization. 22
- 2.5 PixSelect selects reliable keypoints for localization. It abstains from picking features from dynamic objects like cars and pedestrian, changing anatomies like trees, and non-informative regions like the sky. 24
- 3.1 Our method utilizes relative geometric information from adjacent cameras as well as distant cameras to optimize the Deep Network weights for the benefit of global localization based on a single image. The proposed work applies relative pose constraints that are obtained from consecutive cameras along each sequence (denoted as single lines between two successive cameras) and from distant cameras across sequences that are far in the spatio-temporal space of the scene (denoted as double lines between two different sequences). These constraints are applied simultaneously for each training iteration. 26
- 3.2 A diagram of our method. At training time, sequences of images are used to impose relative geometric constraints. For each image that is fed to the Deep Network, the following is predicted: the corresponding 3D coordinates in the global reference frame (Sec. 3.3.2), the depth which is used to estimate the 3D coordinates in the camera frame (Sec. 3.3.3), and weighting factors (Sec. 3.3.6). Rigid alignment is used to align the two 3D point clouds to estimate a pose. L_{3D} , L_D , and L_{pose} are losses that are applied during the training phase. In addition, relative pose errors along and across sequences are utilized as an additional signal to train the Deep Network (Sec. 3.3.5). $\Delta_{i,j}^k$ stands for the relative pose error between two successive cameras i and j in the same sequence k (along sequences). $\Delta_i^{k,l}$ stands for the relative pose error between the frame i of each of the sequences of k and l (across sequences). At inference, an image is fed to the Deep Network to estimate the two map representations. Given that, a pose is obtained through weighted rigid alignment. 30

4.1	Illustration of our visual localization proposal on samples from Cambridge Landmarks (Hospital scene). Our method requires a set of images and the corresponding poses as the only labels for training. Left side: Given a single image, our method estimates the global pose of the camera in a given scene. Right side: we display the intermediate outputs of our proposal, which are used to estimate the pose. For an input image, the proposed pipeline estimates two point clouds and a set of weights. The first point cloud represents the scene geometry (X, Y, Z coordinates) in the camera coordinate frame, while the second point cloud represents the scene geometry in the global coordinate frame. These two point clouds and the predicted weights are used to estimate the camera's global pose. On the right side, we visualize three sample input images, their corresponding indirectly estimated 3D scene representations (point clouds), and the weights. At the top, on the right side of the figure, we can see only one 3D point cloud, which corresponds to three overlaid point clouds in the global coordinate frame, also estimated by our algorithm for the considered sample images. Though our method implicitly estimates 3D point clouds of the scene in local and global reference frames, it is not a mapping or 3D reconstruction algorithm but a localization algorithm that implicitly learns and uses 3D scene geometry.	45
4.2	A diagram of the proposed training method. Given images and the corresponding global pose labels, the network learns to indirectly (i.e., without explicit labels) estimate two 3D point clouds (one in global and the other in camera coordinate systems) and a set of weights. For the 3D coordinates in the camera coordinate system, the network predicts the depth, which is then back-projected to 3D space using Equation. (4.6). A rigid alignment module aligns the two point clouds according to the weights to estimate pose. The loss terms employed for training are visualized in red.	48
4.3	Visual samples of the predictions obtained by the network on samples from 7Scenes. For visualizing the 3D coordinates, we map the X, Y, Z coordinates to RGB values.	56
5.1	We propose iCDC, a method to supervise feature extractors across different domains. Traditionally, correspondences are generated by augmenting and applying homographic transformations to images. iCDC generates accurate correspondences across images captured under different views and visual domains.	62

- 5.2 Joint trajectory map refinement. (a) Frames with registered geolocations along multiple trajectories of the 4Seasons dataset are placed as anchors into a global map. (b) The remaining frames have poses only in a local reference frame optimized through stereo visual-inertial odometry. They are integrated into the map by interpolating them around the anchor frames. (c) SuperPoint [34] features are extracted and matched using SuperGlue [118]. Using the poses in the initialized map, the matched keypoints are triangulated into 3D. (d) Pixel-Perfect-SfM [85] is leveraged to optimize the 3D points and poses in the map through feature-metric bundle adjustment. 68
- 5.3 The figure showcases keypoints and matches from the 4Seasons dataset and from our refined maps. Matches across frames are highlighted in green. For the 4Seasons dataset, keypoints are matched using corresponding depth values and frame poses, while matches from our refined maps are directly extracted from the maps. 68
- 5.4 Keypoint reprojection for pose quality analysis. Each row presents image pairs captured during different seasons. Keypoints from each left-side image are reprojected onto the right-side image using ground-truth poses and keypoint depths. The top and bottom rows are reprojected keypoints from the 4Seasons maps and our maps, respectively. Lines have been drawn to highlight references and corresponding reprojected keypoints. The projected keypoints along the highlighted lines reveal consistent reprojection errors across the 4Seasons frame pairs. These consistent errors indicate inaccurate relative poses. The same can not be observed in the case of our refined maps. 70
- 5.5 Cumulative Distribution of Relative Pose Errors on the 4Seasons Dataset [156]. For each scene, we present the AUC curves separately evaluated against our ground truth poses. Cross-domain plots are marked with filled lines while intra-domain plots are marked with dashed lines. The plots reveal a large performance gap between intra- and cross-domain localization. 73
- 5.6 Overview of explored correspondence methods. Homographic transformations (a) are used to train most extractor methods out of convenience. Dense correspondence estimation networks (b) can estimate correspondences between real-world images across arbitrary viewpoint changes. However, these methods can be inaccurate, particularly across visual domains. Sparse Structure-from-Motion (SfM) map matches (c) can serve as correspondences, providing highly accurate sparse correspondences across domains and views. In our method iCDC (d), we train domain-specific NeRFs using cross-trajectory SfM maps and generate correspondences by comparing the implicit 3D geometries. Highlighted regions are where correspondences are extracted. 76

5.7	The figure presents image pairs illustrating iCDC Correspondences. Fig. (a) shows pixels of correspondences satisfying loop consistency (LC) tests colored according to LC thresholds. Fig. (b) depicts the same mask, except marking pixels as green that pass the LC test but not the depth consistency (DC) test with a 15cm DC error threshold.	79
5.8	Comparison between our NeRF correspondences and WarpC [141] correspondences. Correspondences are generated for an image pair (top row). The second and third rows of the left column show WarpC and iCDC correspondences, respectively. More specifically, the right-hand side image is warped onto the left-hand side image following the correspondences. We zoom in on windows for detailed examination. In the case of accurate correspondences, the warped frames align with the left-hand side frame. .	82
5.9	Cumulative distribution of cross-domain pose errors on the 4Seasons dataset [156] by supervising R2D2. Our trained R2D2 extractors are benchmarked using our ground truth poses (a). Lower and upper bounds (L+U Bounds) are the error distributions of the best cross-domain localization method (SuperPoint [34]) and R2D2 [114] evaluated on intra-domain respectively). Dashed lines are our trained extractors used to benchmark iCDC.	83
5.10	Cumulative distribution of cross-domain pose errors on the 4Seasons dataset [156] by applying our supervision method on SiLK. We complement Fig. 5.9 by plotting the results of our trained SiLK method. Upper bound results are obtained from the best cross-domain localization method (SuperPoint [34]) and R2D2 [114] evaluated on intra-domain, respectively). Dashed lines are our trained extractors used to benchmark iCDC.	84
5.11	Cumulative distribution of cross-domain pose errors on our benchmark dataset. Keypoints are matched using either Euclidean distance-based Nearest Neighbors (NN), or using LightGlue [86]. The lower bound is the SuperPoint [34] with default weights. The upper bound is the intra-domain localization performance of [114] with default weights.	90
5.12	Additional comparison between our NeRF correspondences and WarpC [141] correspondences. Correspondences are generated for an image pair (top row). The left column's second and third rows show WarpC and iCDC correspondences respectively. More specifically, the right-hand side image is warped onto the left-hand side image following the correspondences. We zoom in on windows for detailed examination. In the case of accurate correspondences, the warped frames align with the left-hand side frame. .	93
5.13	Optimized bounding box alignment. The keypoint point cloud is the set of 3D points a map corresponding to a scene block. Initially, the point clouds are scaled. Using Principal Component Analysis, we calculate the point cloud's principal components. To fit as much of the scene into the smallest box possible, we center and transform the point cloud to align the principal components with the body diagonal of the boxes.	95

5.14	Bounding box alignment illustrations. Displayed are scenes aligned within scale 16 bounding boxes. Camera trajectories are depicted as tubes, with the point cloud representing the 3D scene for each camera frame. Frames within blocks are shown as pink point clouds and purple camera frames, while yellow camera frames and green point clouds depict those in the buffer region.	95
6.1	Left: to render the view from a certain camera perspective, 3D points of the scene are projected to the camera image. The resulting 2D projections correspond to 3D points that are seen by the camera (green points) as well as 3D points that are not seen from the current camera view (gray points). Using these projections to render a view results in low-quality images, as shown on the right side. Right: Examples of rendered views together with the actual reference view and 3D scene projections into the image. Both examples show incoherence between the points projections (geometry) and the reference view (appearance). As a result, rendering the view using these projections ends up in deteriorated views. The samples depict the projection of unseen buildings in the current camera view. That is, the obtained projections do not match the actual view as seen by the camera. As a consequence, the renderings are of low quality. In the first sample, incorrect projections propagated the unseen building in the reference view to the rendered view. In the other sample, this mismatch resulted in degraded rendering. The arbitrary views show the buildings which ended up in the projection of the camera.	101
6.2	A sketched representation of our pipeline. A scene is represented by a set of source images and a point cloud. To render the view from a novel pose (yellow camera), a visibility retrieval module retrieves the 3D points that are relevant to the current view. The descriptors of the retrieved points are rasterized into the camera at different downsampled resolutions. Then, a refiner network renders the image by mapping the rasterizations into a novel view. At training time, a multi-resolution discriminator is deployed to improve rendering quality by classifying the generated images as real or fake. \mathbf{f}_i is the descriptor that corresponds to point \mathbf{g}_i . . .	105
6.3	Different proposals to address large point cloud rendering: a) clipping points outside certain threshold, b) clustering of the scene, c) taking closest points (nearest z-buffer) d) Ours: building connectivity relationship between 3D points and poses/images. We compare the different proposals in section 6.3	106
6.4	Rendering results of our method compared to other neural point-based methods.	112
6.5	Using our proposed connectivity relationship drastically improves scene reconstruction using 3D Gaussians Splatting [71]	114
6.6	Our neural point based graphics methods obtains better renderings on Kitti360 than 3D Gaussian splatting [71].	115

Chapter 1

Introduction

1.1 Motivation and Preliminaries

1.1.1 Motivation

Camera localization is essential for many applications in Robotics, virtual/augmented reality, and autonomous driving. While GPS-enabled devices have been, at length, used to locate oneself or to navigate to a certain destination, the limits of GPS-based re-localization become apparent when conducting activity in areas where GPS signals do not work, disappear, or get interrupted, such as indoor areas and cities with tall buildings.

An alternative to GPS-based re-localization, visual camera re-localization locates a camera sensor from the images that it produces. Given an image or set of images, visual re-localization aims to compute the absolute pose of a camera, composed of its location and orientation, relative to a global reference frame of a scene.

Traditionally, computing the absolute pose of a camera has been solved with image retrieval [125] or structure-based methods [121, 122, 123]. Given an image for which a pose is to be computed, image retrieval searches against a database of images with known poses for the closest one or closest K images [113, 53]. A pose is then set to be the pose of the nearest neighbor or computed from the k nearest poses. Image retrieval thus obtains a rough pose estimation. On the other hand, structure-based methods obtain 2D salient features from a query image and search for matches in the reference 3D model of the scene. The set of 2D-3D correspondences is then used to compute a pose using perspective-n-points (PnP) algorithm [47, 48, 77] within a RANSAC scheme [43, 17]. The RANSAC scheme filters the outlier matches. The pose can be further refined from the set of inliers. Compared to image retrieval, Structure-based methods obtain more accurate poses. However, structure-based methods require storing the 3D model of the environment, which can be in the size of Gigabytes for large scenes.

With an indefinite number of correspondences, computing a pose using RANSAC can be expensive.

With the recent advances in computing resources and the quick rise of accompanying open source machine learning frameworks such as Pytorch [106] and Tensorflow [1], deep learning was quickly adopted by the research communities and industries. The past decade has seen marvelous advancements in deep learning, including the emergence of various network architectures and training techniques [38]. Deep learning is seen as the driving force for achieving the goal of artificial intelligence. It has set the footprint for the advancement of many technologies such as computer vision [72, 153, 110], natural language processing [35, 37, 138], and speech recognition [89, 167]. Specifically, deep learning has demonstrated impressive success in the field computer vision by advancing many tasks including image classification [168, 46], object detection [109, 111], image segmentation [87, 56], visual tracking [169, 152], and image restoration [36, 75].

Similarly, deep learning was utilized for camera localization in the early stages [70]. Initial proposals map an input image to an absolute pose by guiding a Deep Network to regress six quantities corresponding to the six degrees of freedom (6 DoF) pose. Utilizing deep learning in this regard has provided few advantages over existing methods. Firstly, a map of the scene, which may require Gigabytes of Memory, is not required to be saved explicitly. Instead, it is encoded implicitly in the weights of a Deep Network, which accounts for tens of Megabytes. Secondly, with graphic processing units, localization can occur in real time, with no additional overhead for matching salient features to a 3D model. However, a primary shortcoming of pose regression is expressed in the accuracy of the estimated poses, which is comparable to image retrieval methods [113, 53].

In this dissertation, we propose solutions to utilize deep learning to improve camera re-localization. Re-localizing a camera assumes that the area has been visited previously (for example, while building the dataset for training). Intuitively, improving an area's mapping directly influences localization and can be seen as a sub-task towards the goal of enhancing localization. In the context of deep learning, this might entail improving the explicit map, such as 3D coordinates, or the implicit representations, such as weights/encoding of the deep network that represents the map.

1.1.2 Formulation

We can define the desired problem as the following: For an input image \mathbf{I} , taken by a camera C_r , an absolute pose estimator is required to compute the 3D location and 3D orientation of camera C_r in a global coordinate system of a given scene. This can be formally defined as:

$$F(\mathbf{I}) = (\mathbf{t}, \mathbf{R}), \tag{1.1}$$

where F is a function that takes as input the image \mathbf{I} and outputs the pose $\mathbf{T} = [\mathbf{t}, \mathbf{R}]$ of Camera C_r . $\mathbf{t} \in \mathbb{R}^3$ is the 3D position of the camera center in the global reference frame of the scene, and \mathbf{R} is the camera orientation relative to the global frame. \mathbf{R} can hold different representations, including rotation matrices, quaternions, and Euler angles. In our derivations, we stick to the rotation matrix representation $\mathbf{R} \in \mathbb{R}^{3 \times 3}$. The input image \mathbf{I} holds the dimension (C, H, W), where C, H, W are the number of channels, height, and width of the image.

1.1.3 Data Preparation

For our data-driven solutions for re-localization, the camera needs to be localized metrically, i.e., in metric units (meters and degrees). Consequently, a metric-labeled dataset is needed. For this purpose, the images and the corresponding absolute poses are the minimal labels required. For building this dataset for outdoor scenes, the absolute poses of the collected images can be computed by structure from motion algorithms (SfM) [127]. Popular SfM tools are COLMAP [127], Bundler[134], and VisualSFM [158]. SfM is a technique that utilizes a series of 2-dimensional images to reconstruct the 3-dimensional structure of a scene or object. SfM can produce a point cloud-based 3-D model. The reconstruction is incremental, in which poses are computed from matched features. These poses can, in return, be used to triangulate more points. Estimated poses and 3D points are regularly optimized with bundle adjustment. On the other hand, indoor scenes usually contain repetitive and textureless areas, which can be challenging for SfM algorithms. Thus, ground-truth camera poses can be obtained for indoor scenes with a camera tracking system like KinectFusion.

1.1.4 Related Work

Visual based localization methods can be categorized into two directions/groups: direct and indirect methods.

Direct methods estimate camera pose deterministically using a fixed pipeline without depending on external processes like retrieval from a database of images, matching features, sampling of pose hypotheses, and refinement of selected hypotheses using matched inlier correspondences. An example of a direct method is pose regression, in which the pose is regressed from a Deep Network [70, 69, 148, 149, 29, 20, 146, 105]. (illustrate with a diagram)

Indirect methods are methods whose pose estimation consists of more than one step, like obtaining many pose hypotheses for further processing/refinement and/or condition on external dependencies like image retrieval. Compared to direct methods, they include additional probabilistic components such as database querying, hypotheses sampling, pose refinement, outliers filtering by RANSAC [43], etc. Methods that look for 2D salient features and their 2D or 3D matches for hypotheses sampling-based pose estimation are indirect pose estimation methods [123, 15, 16, 18, 119, 3].

Indirect methods obtain more accurate poses than direct methods.

In this dissertation, we propose data-driven solutions to improve localization. We introduce novel geometric constraints into the learning paradigm in order to improve the mapping representation of the environment in its implicit and explicit forms and increase the localization accuracy. Besides, we propose utilizing generative schemes in novel ways to generate source data for improving mapping and localization. Concretely, we propose, in Chapter 2, an indirect method that selects discriminative features from the map that are useful for localization and utilizes them and their estimated 3D correspondences to obtain a pose. The proposed solution goes past sources of outliers to select reliable correspondences for accurate and efficient localization. In Chapter 3, we propose using additional geometric constraints obtained from the available labels without any extra effort in data collection to improve direct localization. These constraints are spatio-temporal relative poses that are computed from adjacent and distant cameras of the scene. In Chapter 4, we utilize pairs of image-pose data only to estimate the 3D map of the scene without explicit ground-truth labels of the map. Given a single image at inference, the proposed deep network estimates two map representations and aligns them geometrically to improve the accuracy of estimated poses. In Chapter 5, we go a stride further in tackling long-term localization by utilizing implicit volumetric scene representations (NeRF [96]) to generate ground-truth correspondences. These reliable correspondences are utilized to improve localization across long-term illumination changes. Here, we also contribute with a benchmark for evaluating different keypoints detection and description methods for mapping and localization. In Chapter 6, we propose a novel point-based generative model to synthesize novel views in the map.

The next section provides a more detailed overview of the dissertation outline, research questions, and contributions.

1.2 Outline, Research Questions and Contributions

In this section, we summarize the content of each chapter of this thesis by presenting the research questions and how we tackled them.

1.2.1 Chapter 2: Learning Geometric Prior About Reliable Pixels for Absolute Localization

Research Question: An estimated 3D map is infiltrated with noisy and unreliable measurements. Can a geometric prior be learned to inform about reliable and non-reliable regions of a scene for pose estimation? Is every part/pixel of an image important for absolute localization?

Previous deep learning-based localization methods used the whole scenery part, which is seen from a certain perspective, to estimate a pose. In this work, we propose

to learn a geometric prior to inform about where reliable and discriminative regions are in a given scene and use correspondences from these regions to estimate a pose. Our proposal, PixSelect, recognizes unreliable image regions such as the sky, trees, and pushes, moving objects like cars and pedestrians in addition to occluded areas. Consequently, it avoids selecting correspondences from these regions, allowing for a controllable selection of reliable correspondences for localization. Our experiments show that utilizing our learned geometric prior allows low and reliable correspondences to be selected, enhancing localization accuracy and efficiency.

1.2.2 Chapter 3: Spatio-Temporal Constraints Across Distant Cameras for Absolute Localization

Research Question: The problem at hand is a geometric problem. How can we utilize the commonly available labels, image-pose pairs, to create novel geometric constraints? And how can we incorporate these constraints for the benefit of localization?

With the rise of data-driven approaches, recent methods have utilized the absolute pose labels of images of a given scene to guide a network to relocalize an image in that scene. In this work, we create novel additional signals for training a relocalization pipeline by exploiting the absolute poses to generate constraints in the form of relative poses. These relative geometric constraints are computed not only from camera poses that are close in space and time but also from cameras that are distant in the space and time of a given scene. With that, the deep network, which represents the map, uses additional training signals to encode more geometric information of a scene in its weights. Given a single image at inference, the localization pipeline uses the deep network that encodes these additional constraints to obtain a more accurate pose.

1.2.3 Chapter 4: Learning Scene Geometry without Direct Labels for Absolute Localization

Research Question: How can we utilize the available absolute poses to learn geometric attributes of a given scene without direct supervision, which eventually can be used to estimate a pose?

The incorporation of geometric information is necessary for improving the accuracy of pose estimation. Previous works propose to utilize a network to regress geometric information, such as 3D coordinates of the image pixels in a given reference frame. This geometric information is then used to compute a pose. For learning these geometric attributes, the methods use additional ground-truth 3D coordinates to supervise the training. In this chapter, we propose learning geometric information from a scene by relying only on the available, absolute poses of a scene. We employ a parameter-free and differentiable rigid alignment module to guide a deep network in estimating a scene representation in two reference coordinate systems by aligning them to match

the pose label. At inference time, the deep network estimates, for the inputted single image, two explicit representations of the scene in the form of point clouds and aligns them geometrically to increase the accuracy of estimated poses.

1.2.4 Chapter 5: Cross-Domain Correspondences for Long-Term Localization

Research Question: How can we make localization robust to long-term illumination changes? What features of a given environment can be used to address localization across different day-time, seasonal, and weather changes?

Localization methods commonly target environments that share the same illumination conditions as the mapping environment. However, the localization accuracy drops when localizing under different daytime, seasonal, or weather conditions. This chapter investigates the impact of such long-term illumination changes on localization accuracy. We thus develop a benchmark to evaluate keypoints detection and description methods for localization under long-term illumination changes. Accordingly, we develop a novel approach to generate accurate and reliable correspondences across different illumination conditions to tackle the gap in localization. We propose to represent the same environment with different implicit representations (based on neural radiance fields), each corresponding to a different illumination condition. Following along, the proposed method utilizes the underlying 3D geometry of these representations to exchange correspondences and keep the reliable ones. We use these correspondences to enhance localization under different illumination conditions.

1.2.5 Chapter 6: Point-based Large-Scale Novel View Synthesis

Research Question: Data sparsity limits the accuracy and generalization capability of direct localization methods. How can we generate additional image-pose pairs, ensuring dense scene coverage?

Data is vital for data-centric approaches. However, driving around and collecting data that densely covers a scene for the sake of mapping and localization can be expensive. In this chapter, we propose a novel point-based generative pipeline that is scalable and fast to synthesize novel views. The proposed pipeline utilizes a point cloud of a scene and a set of source images. It associates the 3D points with neural descriptors that are learned to encode appearance and geometric information. It builds a connectivity relationship graph between the images (appearance) and the 3D points (geometry) to overcome their mismatch. This mismatch originates from incompatibility between what the camera sees (appearance) and what the geometry informs about the scene (geometry), which leads to degraded rendering quality in large scenes. To synthesize an image from a novel view (i.e., create image-pose pairs), the connectivity relationship is queried to retrieve the 3D points that are seen from this perspective and

rasterized into the image. This learning scheme is guided by image reconstruction loss and adversarial training setup.

1.3 List of Publications

This dissertation includes material that has been published in papers organized as follows:

- Chapter 2: **Mohammad Altillawi**, “PixSelect: Less but Reliable Pixels for Accurate and Efficient Localization”, 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 2022, pp. 4156-4162.
- Chapter 3: **Mohammad Altillawi**, Zador Pataki, Shile Li and Ziyuan Liu, “Global Localization: Utilizing Relative Spatio-Temporal Geometric Constraints from Adjacent and Distant Cameras”, 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, MI, USA, 2023, pp. 3358-3365.
- Chapter 4: **Mohammad Altillawi**, Shile Li, Sai Manoj Prakhya, Ziyuan Liu and Joan Serrat, “Implicit Learning of Scene Geometry From Poses for Global Localization”, in IEEE Robotics and Automation Letters, vol. 9, no. 2, pp. 955-962, Feb. 2024.
- Chapter 5: Zador Pataki, **Mohammad Altillawi**, Menelaos Kanakis, Remi Pautrat, Fengyi Shen, Ziyuan Liu, Luc Van Gool and Marc Pollefeys, “Long-Term Invariant Local Features via Implicit Cross-Domain Correspondences”, Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) in Nov. 2023. Available on arXiv preprint arXiv:2311.03345, Nov. 2023.
- Chapter 6: **Mohammad Altillawi**, Sai Manoj Prakhya, Fengyi Shen, Shile Li, Jin Xin and Ziyuan Liu, “Large-Scale Novel View Synthesis with Enhanced Neural Point-Based Graphics”, Submitted to IEEE Robotics and Automation Letters (RA-L) in July 2024. It will be available on arXiv.

The published papers have been modified to conform to the style and requirements of this dissertation. I want to thank my co-authors and the publishers for permitting the use of this material.

1.4 List of Patents

- **Mohammad Altillawi**, Ibrahim Hallfaoui, Onay Urfalioglu, “Data processing apparatus and method for determining a pose”. An application for a patent is submitted on 10.09.2021 with the number PCT/EP2021/074880. Patent is filed worldwide and in the USA. Accessible at: <https://patentimages.storage.googleapis.com/76/02/cf/68dedad8eefc2e/WO2023036431A1.pdf>

- **Mohammad Altillawi**, Shile Li, Ziyuan Liu, “Method for obtaining the global camera pose from a single image by aligning the implicitly predicted depth and 3D global coordinates which are learned from pose labels only”. An application for patent filing with application number PCT/EP2023/055749 is submitted on 07.03.2023 to the European patent Office.
- **Mohammad Altillawi**, Ziyuan Liu, “Method for obtaining camera pose by utilizing relative pose constraints from adjacent and distant cameras covering the spatio-temporal space of a scene”. An application for patent filing with application number PCT/EP2023/076871 is submitted on 28.09.2023 to the European patent Office.

Chapter 2

PixSelect: Less but Reliable Pixels for Accurate and Efficient Localization[†]

Accurate camera pose estimation is a fundamental requirement for numerous applications, such as autonomous driving, mobile robotics, and augmented reality. This work addresses the problem of estimating the global six degree-of-freedom camera pose from a single RGB image in a given environment. Previous works consider every part of the image valuable for localization. However, many image regions, such as the sky, occlusions, and repetitive non-distinguishable patterns, cannot be utilized for localization. In addition to adding unnecessary computation efforts, extracting and matching features from such regions produces many wrong matches, which in turn degrades localization accuracy and efficiency. PixSelect addresses this particular issue and shows that by exploiting an interesting concept of sparse 3D models, we can exploit discriminatory environment parts and avoid useless image regions for the sake of a single image localization. Our work acts naturally as an outlier filter by avoiding selecting keypoints from non-reliable image regions such as trees, bushes, cars, pedestrians, and occlusions. Besides, obtaining reliable pixels for localization, it estimates the corresponding 3D global coordinates for each pixel. By doing so, it obtains the 2D-3D correspondences directly, i.e., without matching, from the input image. By circumventing these sources of outliers, PixSelect chooses a relatively low number of correspondences, making it efficient and accurate. Choosing as few as 100 correspondences surpasses similar methods that localize from thousands of correspondences while being faster. Our work exceeds state-of-the-art methods on the outdoor Cambridge Landmarks dataset. Only by relying on a single image at inference does it outstand direct pose

[†]This chapter is based on the work "PixSelect: Less but Reliable Pixels for Accurate and Efficient Localization", published in the International Conference on Robotics and Automation (ICRA), pp. 4156-4162, 2022. The article has been modified to fit the format and requirements of this dissertation.

regressors, even those that learn from the sequence of images. In particular, compared to these methods, it achieves an improvement of localization by 33% on Old-Hospital scene.

2.1 Introduction

Camera localization has enabled robots and cars to navigate in areas where other localization sensors, such as GPS, may fail. However, due to delays or blockages in satellite signals, GPS becomes less reliable, affecting localization accuracy. Camera-based global localization has been addressed by classical structure-based methods [121, 122, 123] as well as by deep learning-based approaches [70, 69, 148, 20, 149, 105, 29, 146, 15, 16, 18].

Classic structure-based methods [121, 122, 123] estimate a pose by extracting 2D features from query images and then matching these features to the 3D points in a 3D scene model. From the obtained set of 2D-3D correspondences, a pose is computed using a Perspective-n-Points (PnP) algorithm [47, 77, 48] in a RANSAC [43] scheme, which filters outlier correspondences. Structure-based methods typically obtain a fine pose localization. However, in online deployment, they explicitly require a 3D model, which adds a memory burden that increases with the expansion of the localization area. The process of matching descriptors to obtain correspondences can be expensive and time-consuming. In addition, the obtained correspondences are often noisy. Consequently, the number of outliers rises with the model's growth, imposing additional run-time demands for the RANSAC filtering scheme [43] at the cost of localization run-time.

The recent advances in deep learning have inspired many works to utilize these advances for the benefit of localization. One direction of these works is to deploy a deep network to map input image or sequence of images to a pose [70, 69, 148, 20, 149, 105, 29, 146]. The common practice of these direct pose regressors is to encode the image in a latent feature vector. Following that, regression layers map the latent encoding to a pose as two separate entities: camera position and camera orientation. Usually, learning this direct mapping starts by transferring knowledge, which is learned from a large classification dataset, to initialize the network weights for the downstream task of pose estimation. The problem of pose estimation is then reduced to finetuning the weights using the image-pose pairs for the task of pose regression. The proposals of this art aim to constrain the latent feature vector to encode helpful information from the image or stream of images for the benefit of localization. Unlike structure-based methods, direct pose regression methods are faster and have a lower memory footprint. However, their localization accuracy is much lower.

Following the advances of deep learning, some works [132, 15, 16, 18] took an approach different from direct pose regressors. Instead of regressing the pose, the 3D

scene is regressed to form a set of 2D-3D correspondences out of which a pose is calculated. In these works, every part of the image is considered valuable for localization, and correspondences are obtained from every grid of the image. These works achieve the state of art localization results.

Our approach compromises a hybrid method that deploys deep learning with the classic geometric algorithm for pose estimation. In particular, we utilize deep learning to learn the 3D scene, specifically the discriminatory parts of the scene, and obtain the 2D-3D correspondences directly. For pose estimation, we employ the well-established classic geometric algorithms: the PnP algorithm [47, 77] and RANSAC [43] scheme.

The well-established classic algorithms work off-the-shelf and independent of the scene. Learning these algorithms, instead, may overfit the learned model to the specific learned scene content and can hinder its generalizability to unseen scenery. Rather than utilizing deep learning to learn PnP or RANSAC, we keep the geometric pose estimation to the classic algorithms. Instead, we exploit deep learning to learn suitable features for localization. Following along, PixSelect employs deep learning to learn two tasks from a single image simultaneously. The first is the 3D scene, where the camera is to be localized. The second is where to select features that are beneficial for localization. Thus, we do not consider every image region valuable for localization. We avoid regions like the sky, clouds, repetitive patterns such as walls, ground, pavements, grass spaces, trees, and occluded areas. Refer to Fig. 2.1 for examples of our learned selection of reliable image regions.

Our main contributions can be summarized as follows:

- We present a new hybrid camera re-localization pipeline that combines deep learning for learning the 3D scene and good localization features and the classic geometric algorithms for estimating the global pose. The proposed approach acts by nature as an outlier filter. It is efficient, accurate, and surpasses state-of-the-art methods on the Cambridge Landmarks dataset [70].
- We present the first approach that learns 2D keypoints and 3D scene coordinates in one framework.
- We present a keypoints detector that can avoid selection from nondiscriminatory and occluded areas.

2.2 Related Work

In the following, we discuss the related work for solving camera global re-localization.

Pose Regression: Pose regressors learn a mapping function from either a single input image [70, 69, 148] or a sequence of images [29, 20, 146, 149, 105] into 6 degree-of-freedom (Dof) Pose. A network encodes the input into a latent feature, out of which the pose is regressed. Different learning strategies are followed to constrain the latent vector either by LSTMs [148], relative pose information [29, 20, 146, 105], 3D model [69] or



Figure 2.1: Not all image regions are equally important for visual localization. Our method, PixSelect, selects 2D keypoints from discriminatory image regions that are useful for localization. The left side images show the valid keypoints obtained by PixSelect for the sake of localization. The right side images are the corresponding predicted heatmaps, with good image regions highlighted in yellowish-green (best seen in colors). Our method can emphasize reliable regions for localization and avoid other nonrelevant regions such as the trees, the cloudy sky, and repetitive plane structures like walls, in addition to pedestrians and cars. We draw your attention, in particular, to the bottom image and the corresponding heatmap, where it can be observed that our method is able to distinguish the trees from the structure (building). While emphasizing the background building, it does not select keypoints from the tree branches. It also ignores the reflective rounded windows.

attention [149]. As the mapping is direct, the execution time is fast. However, the common practice is that these methods model the geometric pose estimation problem as a regression task. In contrast, we keep the pose estimation task to the classic algorithms [47, 77] and argue that through pose regression, 3D geometric information cannot be adequately utilized for pose estimation, resulting in inaccurate poses.

Sparse feature matching/alignment: Active Search [123] provides a prioritization

scheme to establish 2D-3D matches and terminates correspondence search once enough matches are found. On that, our work obtains matches directly by regressing for the pixels of interest, their 3D coordinates in a global reference frame, bypassing the computationally expensive step of descriptors matching. Furthermore, Active Search requires the reference 3D model of the target scene at inference. By learning the scene, our proposal predicts the scene at inference, avoiding the need for a model that demands more memory with an increased target area for localization. Saving only the weights of the deep network keeps a constant memory footprint. PixLoc [119] utilizes a query image, reference images, pose priors, and a reference 3D SfM model to learn good features for localization. PixLoc [119] can improve sparse feature matching by refining keypoints and poses. It localizes in a large environment given a pose prior. From a simpler training method, our proposal learns better features for localization. Their work recognizes the sky and clouds and repetitive nondiscriminatory patterns as useful for localization. In contrast, these regions are down-weighted by our work. Without relying on reference 3D models or pose priors, our work obtains lower localization median errors on the Cambridge landmarks dataset [70] while being more efficient. PixLoc [119] extracts features from an image in 100 ms and may require up to one second to localize an image [119].

While our keypoints detector could be paired with off-the-shelf descriptors (hand-crafted or learned) and utilized for reconstructing the environment and for sparse feature matching-based localization, we explore this direction in future works. We focus in this work on global camera re-localization from a single image.

Scene Regression: Rather than utilizing deep learning to learn pose regression from image-encoded features, other works similar to ours learn the 3D scene to form a set of 2D-3D correspondences for localization. DSAC [15] proposed a method to make RANSAC differentiable and applied the work on camera localization. The differentiable RANSAC proposed by DSAC is specific to the dataset and does not generalize to other scenes. In contrast to learned RANSAC, off-the-shelf RANSAC [43] frameworks, which we put to use, are not tight to a specific scene. Two followup works, DSAC++ [16] and DSAC* [18] improved, among other things, the localization accuracy of DSAC [15]. They consider every pixel of the image valuable for localization. Thus, they select thousands of correspondences to estimate a reliable pose hypothesis. In contrast to DSAC and followers, our work avoids selecting pixels from non-discriminatory image regions and occlusions. Our work estimates the reliability of an image pixel for localization. This allows our work to avoid areas that are sources of outliers, prioritize correspondences, and select a lower number of keypoints.

2.3 Method

Overview: The proposed work, PixSelect, employs deep learning to simultaneously select the relevant 2D keypoints from the input single image and estimate the corresponding 3D global coordinates. The selected set of 2D-3D correspondences are

passed to a PnP solver [47] in a RANSAC [43] framework to obtain a pose hypothesis. The proposed work is sketched in Fig. 2.2. On one side, mapping the pixels of the image to 3D coordinates enables our method to form direct correspondences and avoid the expensive step of matching features from an image to a 3D model. On the other side, not all pixels of a given image are reliable and suitable for localization. Thus, the proposed method selects the 2D-3D correspondences estimated to be distinguishable while accounting for choosing a minimalistic set of keypoints for efficient and accurate localization.

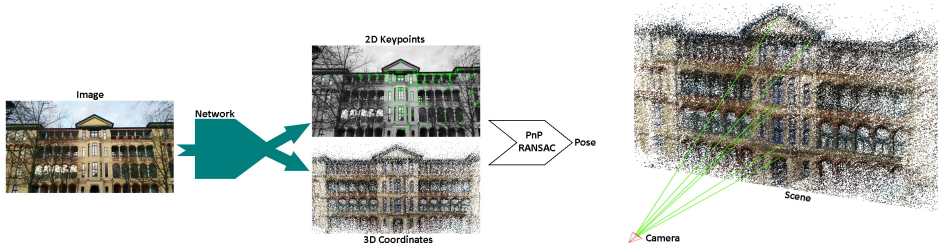


Figure 2.2: A sketched representation of our pipeline: The processing flow is left to right. For a given input image, a deep network produces two outputs. The first is a set of reliable 2D keypoints. Here, we already show the valid 2D keypoints that are overlayed on the image. The second is a set of 3D coordinates relative to a global coordinate system. Here, we show an illustrative example. Correspondences are obtained directly by selecting the 3D coordinates corresponding to the valid 2D keypoints. The set of selected 2D-3D correspondences is then used to obtain a pose using Perspective-n-Point [47] in a RANSAC scheme [43]. The most right sketch shows the pose computation step of the camera relative to a scene where the green lines describe the 2D-3D correspondences. For a description of the network architecture, refer to section 4.4.2.

Motivation: Many regions of a query image, such as the very far sky and clouds, repetitive and non-discriminatory regions like aliased walls, pavements, streets, trees, and bushes, in addition to occlusions, are irrelevant for localization. In the context of 3D scene regression, these repetitive patterns look the same but belong to different environment locations with different target 3D coordinates. Learning the 3D coordinates of these regions confuses the network and may result in improper predictions. In the context of image-to-image or image-to-3D model feature matching, these regions possess significant sources of outliers. As the image patches of these regions look similar, so are the descriptors of these patches. Matching keypoints from the mentioned regions is usually an added effort, which results in few matches and a lot of outliers. Similarly are the occluding areas. Avoiding these regions reduces the outliers and makes localization efficient and more accurate, as shown in sections 2.4.5 and 2.4.6.

2.3.1 Localization Pipeline

The 2D keypoints: The network estimates a heatmap to infer discriminative regions of the image (refer to Fig. 2.1 for visualizations of samples of predicted heatmaps).

With different confidence values for each pixel, those with the highest activation are selected with their corresponding 3D coordinates for pose estimation. We exploit a practical concept from sparse 3D models to select reliable keypoints from a single image. A sparse 3D model can be created by a structure-from-motion method (SfM) [127] by triangulating the matched keypoints from different views. The idea is that since these keypoints are already matched from different views and have passed the verification steps of SfM, they most likely belong to discriminatory regions in the image and are reliable. To the best of our knowledge, our work is the first to exploit the concept of triangulated features from structure-from-motion to directly learn reliable and discriminatory regions from a single image (Fig. 2.1).

In this context, the network learns to predict a heatmap from a single image. The most activated pixels of the heatmap correspond to projections from a sparse 3D model. That is, given a 3D model, for every training image \mathbf{I} , a reference heatmap \mathbf{Z} is created by projecting the visible 3D points (that are observed from the camera with a pose corresponding to image \mathbf{I}) into the image. The projected pixels are assigned probabilities of one in the reference heatmap. The 2D keypoints branch is trained to output a predicted heatmap $\hat{\mathbf{Z}}$ that matches the target one \mathbf{Z} by maximizing the cosine similarity between them. For an input image \mathbf{I} of size $H \times W$, the loss can be formulated as

$$L_{sim} = 1 - \frac{1}{|\mathbf{P}|} \sum_{\mathbf{p} \in P} \frac{\hat{\mathbf{Z}}[\mathbf{p}] \cdot \mathbf{Z}[\mathbf{p}]}{\|\hat{\mathbf{z}}[\mathbf{p}]\| \cdot \|\mathbf{z}[\mathbf{p}]\|}, \quad (2.1)$$

where $P = \{\mathbf{p}\}$ is the set of the $N \times N$ corresponding patches between \mathbf{Z} and $\hat{\mathbf{Z}}$, $\mathbf{z}[\mathbf{p}] \in \mathbb{R}^{N^2}$ is the flattened $N \times N$ patch \mathbf{p} extracted from \mathbf{Z} ; the same applies to $\hat{\mathbf{z}}[\mathbf{p}]$.

The 3D coordinates: The network estimates the 3D coordinates of the pixels relative to a global coordinate system. However, only the pixels chosen from the 2D keypoints heatmap are used to learn the 3D scene. That is, the loss is applied only to the selected set of 2D Keypoints and corresponding 3D points. Two losses are applied to learn the 3D global coordinates. The first minimizes the difference between the selected 2D keypoints and the corresponding projected 3D points. This is defined as:

$$L_{rep} = \frac{1}{|M|} \sum_i^M \|s_i - \pi(\mathbf{K}, \mathbf{t}, \mathbf{R}, \hat{\mathbf{g}}_i)\|_2, \quad (2.2)$$

where $\hat{G} = \{\hat{\mathbf{g}}_i, \dots, \hat{\mathbf{g}}_M\}$ is the set of the predicted 3D coordinates that directly corresponds to the set of selected 2D keypoints $S = \{s_i, \dots, s_M\}$, M is the number of selected correspondences, π is the projection function that projects the estimated 3D point $\hat{\mathbf{g}}_i$ into image \mathbf{I} using the intrinsic parameters matrix \mathbf{K} , the ground-truth extrinsic: translation \mathbf{t} , rotation \mathbf{R} of camera \mathbf{I} . The second loss minimizes the difference between the predicted and reference 3D scene coordinates. This is formulated as:

$$L_{3D} = \frac{1}{|M|} \sum_i^M \|\hat{\mathbf{g}}_i - \mathbf{g}_i\|_2, \quad (2.3)$$

where $G = \{\mathbf{g}_i, \dots, \mathbf{g}_M\}$ is the set of the ground-truth 3D coordinates that corresponds to the set of predicted 3D coordinates $\hat{G} = \{\hat{\mathbf{g}}_i, \dots, \hat{\mathbf{g}}_M\}$. The total loss is then the weighted sum of the three mentioned losses:

$$L_{all} = \lambda_{sim} L_{sim} + \lambda_{rep} L_{rep} + \lambda_{3D} L_{3D}, \quad (2.4)$$

where λ_{sim} , λ_{rep} , and λ_{3D} are the weighting factors for the loss terms.

At inference, PixSelect expects a single image as input and outputs reliable 2D key-points and their corresponding 3D coordinates. The 2D-3D correspondences are fed into a PnP-RANSAC module to compute the camera pose in the global frame.

2.4 Experiments and Evaluation

2.4.1 Datasets

Following previous works, we conduct experiments on the Cambridge Landmarks dataset [70]. Cambridge landmarks is an outdoor relocalization dataset that contains RGB images of six large scenes, each covers a landmark of several hundred or thousand square meters in Cambridge, Uk. The provided reference poses are reconstructed from structure from motion. The authors provide the train and test splits. We use the provided SfM models to obtain the ground-truth 3D points for each image. Following previous works [16, 18, 119], we do not conduct experiments on the sixth scene, the street landmark, as the provided reconstruction is of poor quality. For the StMarysChurch scene, we removed some images from the training set, including trees or bushes with few reference 3D points.

2.4.2 Architecture and Setup

The architecture is built of a down-sampling encoder, from which two up-sampling branches stem. To illustrate, we adapt the network architecture of DispNet [93]. We keep the contractive part of DispNet and replicate the expanding part to have two branches. In addition, we remove the side predictions and keep a final layer for each expanding branch. The 3D coordinates branch outputs three channels, each for one of the X , Y , Z coordinates. The 2D keypoints branch outputs a single-channel heatmap. The output is normalized to be in the range $[0, 1]$. We use the ReLU [103] as nonlinearity activation for the 2D keypoints branch and ELU [30] for the non-linearity between the 3D coordinates branch layers.

The proposed architecture is modular in that each branch can act as a standalone approach. Each branch can be trained separately and then combined in one training setup, or the whole architecture can be trained together from scratch. Since both branches are trained from the same signal (set of 3D points) and to facilitate the training and avoid hard finetuning, we first train the 3D branch with the L_{3D} loss for 400k iterations. After that, the additional losses can be introduced for roughly 200k iterations. For this setting, the weighting factors are 1, 0.1, and 1 for λ_{sim} , λ_{rep} , and λ_{3D} , respectively. We use the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$, weight decay of 5×10^{-4} , and a learning rate of 10^{-4} .

Input images are rescaled to 480 px height and normalized by mean and standard deviation. During training, data augmentations are applied on the fly. This includes color jittering, random scaling in the range $[2/3, 3/2]$, and random in-plane rotation in the range $[-30^\circ, 30^\circ]$. The ground-truth poses are adjusted accordingly. We apply non-maximum suppression for keypoints selection and choose keypoints with confidence scores higher than a threshold and, respectively, their corresponding 3D coordinates. We use the PnP-RANSAC implementation of OpenCV [19] for pose estimation.

2.4.3 Baselines

We compare our work against the deep learning pose regressors [69, 146] approaches that exploit single as well as sequences of images. We show that methods that compute the pose based on classic algorithms such as PnP and RANSAC exploit the scenery and its geometry better than the pose regressors. We compare our work against the works that exploited the 3D scene geometry. Specifically, we compare against the 3D scene learning methods of DSAC++ [16] and DSAC* [18] and show that we obtain lower localization error from a much smaller number of correspondences. Furthermore, we put our work in comparison against works that exploited the reference 3D SfM model, including the notable classic structure-based method, the Active Search [123] and the recent work of PixLoc [119] that also needs pose priors and show that, without exploiting the ground-truth 3D point cloud of the scene or pose priors, that our method obtains lower localization errors and runs faster. We further compare the proposed method against Neural guided RANSAC [17] and show that we get comparable or lower localization errors, indicating that our obtained correspondences include a low number of outliers. With an ablation study, we further prove that our proposal acts naturally as an outlier filter. To list the results of the baselines, we report the results mentioned in their publications. For ActiveSearch [123], we list the results reported by [69, 16, 18].

2.4.4 Comparison of Localization Errors Against State of Art Methods

Tab. 2.2 reports median localization errors, as translation and rotation errors, on the Cambridge landmarks dataset. Compared to DSAC++ [16] and DSAC* [18], the proposed work obtains lower median translation error on King’s College, Old Hospital, and St Mary’s Church and comparable results on Shop Facade. DSAC++ [16] and DSAC* [18]

learn the 3D scene and pass a few thousands of 2D-3D correspondences to a differentiable RANSAC to estimate the pose. In our work, we obtain a much smaller number of correspondences. While the mentioned works select keypoints that cover the whole image, our selection is more cautious and avoids useless areas and occlusions. Figures 2.1 and 2.3 show samples of selected keypoints by PixSelect. The number of chosen keypoints depends on how discriminatory the image regions are. The more non-discriminatory areas (sky, similar repetitive patterns such as road, green spaces, etc.) in the image, the fewer selected correspondences. These are, on average, 600, 250, 130, and 500 correspondences obtained from King’s College, Old Hospital, Shop Facade, and St Mary’s Church, respectively. Our method was able to localize every frame of the dataset.

Table 2.1: Comparison against State-of-the-art visual localization methods on the Cambridge Landmarks dataset[70]. We report the median translation and rotation errors.

Methods	Cambridge			
	College	Hospital	Shop	Church
Regression				
PoseNetGeo [69]	0.88m, 1.04°	3.20m, 3.29°	0.88m, 3.78°	1.57m, 3.32°
VLocNet [146]	0.84m, 1.42°	1.08m, 2.41°	0.59m, 3.53°	0.63m, 3.91°
Differentiable RANSAC				
DSAC ++ [16]	0.18m, 0.3°	0.20m, 0.3°	0.06m, 0.30°	0.13m, 0.4°
DSAC* [18]	0.15m, 0.3°	0.21m, 0.4°	0.05m , 0.3°	0.13m, 0.4°
3D Structure-Based				
ActiveSearch [123]	0.42m, 0.55°	0.44m, 1.01°	0.12m, 0.40°	0.19m, 0.54°
PixLoc [119]	0.14m, 0.24°	0.16m, 0.32°	0.05m, 0.23°	0.10m, 0.34°
Ours	0.14m , 0.34°	0.14m , 0.5°	0.06m, 0.5°	0.09m , 0.46°

Table 2.2: Comparison against State-of-the-art visual localization methods on the Cambridge Landmarks dataset[70]. We report the median translation and rotation errors.

Methods	Cambridge				
	College	Hospital	Shop	Church	Average
Direct Methods					
PoseNetGeo [69]	0.88m, 1.04°	3.20m, 3.29°	0.88m, 3.78°	1.57m, 3.32°	0.79m, 2.82°
VLocNet [146]	0.84m, 1.42°	1.08m, 2.41°	0.59m, 3.53°	0.63m, 3.91°	
Indirect Methods					
DSAC ++ [16]	0.18m, 0.3°	0.20m, 0.3°	0.06m, 0.30°	0.13m, 0.4°	0.14m, 0.32°
DSAC* [18]	0.15m, 0.3°	0.21m, 0.4°	0.05m , 0.3°	0.13m, 0.4°	0.14m, 0.35°
ActiveSearch [123]	0.42m, 0.55°	0.44m, 1.01°	0.12m, 0.40°	0.19m, 0.54°	0.29m, 0.63°
PixLoc [119]	0.14m, 0.24°	0.16m, 0.32°	0.05m, 0.23°	0.10m, 0.34°	0.11m, 0.28°
Ours	0.14m , 0.34°	0.14m , 0.5°	0.06m, 0.5°	0.09m , 0.46°	0.107m , 0.45°

Similarly, our work obtains lower translation error than ActiveSearch [123] and lower or comparable to PixLoc [119] results. Both works localize against a ground-truth 3D SfM model, which we don't utilize at inference. PixLoc also requires a pose prior.

We further include results of pose regressors that learn from a single image (PoseNet-Geo [69]) or sequence of images (VlocNet [146]). Compared to Pose regressors, our method obtains much lower localization errors. PoseNetGeo [69] incorporates 3D geometric information of the scene in the loss as an additional constraint for learning features suitable for localization. The work, however, treats the geometric pose estimation as a regression problem. VlocNet [146] benefits from learning from sequences of images to constrain the pose regression and obtains lower localization errors than PoseNetGeo [69]. However, the localization errors are still higher than other works which exploit, in some form or another, 3D information for pose estimation at inference like DSAC++ [16], DSAC* [18], ActiveSearch [123], PixLoc [119], and ours. Specifically, works that treat the pose geometric estimation problem using classic computer vision methods that rely on 2D-3D correspondences obtain more accurate pose estimation than a pose by regression. This comparison partly motivated us to get the pose using classic algorithms and confirms the observations of [120] by further comparing against new works.

2.4.5 PixSelect as Outlier Filter

We compare our work against Neural guided RANSAC, NG-RANSAC [17]. NG-RANSAC presents an extension to the classic RANSAC algorithm by training a network to guide RANSAC hypothesis sampling through which outliers are low-weighted while inliers are predicted with higher weights. They evaluate the work on Cambridge scenes for camera localization by combining the architecture of DSAC* [18] for scene regression with NG-RANSAC [17] for guided hypothesis sampling. Tab. 2.3 lists the localization errors of NG-RANSAC and our work. By utilizing reliable keypoints, we obtain comparable results. Specifically, we achieve the best localization on the Hospital scene. In this scene, our work succeeds in avoiding selecting correspondences that fall in occluded areas, cars, trees, bushes, repetitive plane areas, and reflective windows. This emphasizes that significant sources of outliers are avoided by learning to avoid these areas; thus, the localization accuracy is improved.

2.4.6 Importance of Selecting Reliable Keypoints

The proposed method localizes by selecting a set of distinguishable correspondences. The selected keypoints lie in discriminatory parts of the image. This step helps select a minimalistic set of correspondences while incurring low outliers. To show the importance of the selected keypoints, we compare the pose localization errors by selecting two sets of correspondences; each has 200 correspondences. The first set is selected from the points predicted with high scores (above 0.7), while the other set is composed of points suppressed by the non-maximum suppression (predicted with a

Table 2.3: Median position errors on Cambridge Landmarks of our work against Neural guided RANSAC [17]. The results show that the correspondences obtained by our work incur a low number of outliers. It acts naturally as an outlier filter.

Methods	Cambridge			
	College	Hospital	Shop	Church
Neural Guided RANSAC [17]	0.126m	0.219m	0.056m	0.098m
Ours	0.137m	0.14m	0.06m	0.093m

score of 0.4). We run OpenCV PnP-RANSAC [19] implementation with 100 iterations and a re-projection error of 3 px. Results are listed in Tab. 2.4.

Table 2.4: Localization median errors for two sets of correspondences. Set 1: The 200 correspondences of high confidence. Set 2: 200 correspondences with lower confidence. Refer to Fig. 2.3 for visual feedback.

Scene	Set 1	Set 2
College	0.15m, 0.44°	0.43m, 1.5°
Hospital	0.15m, 0.5°	0.45m, 1.6°

Furthermore, Fig. 2.3 provides visualization for the selected keypoints of the two sets.

As can be observed, the set of keypoints that our method selects belong to reliable areas. In most cases, trees, similarly looking pavements, streets, reflective windows, and sky are avoided. This helps the network learn the scene corresponding to the reliable regions and predict the corresponding 3D coordinates. In contrast, the already mentioned areas make it hard for the network to predict the 3D scene reliably. This, in return, degrades the localization accuracy. These results, when compared to DSAC++ and DSAC* (Tab. 2.2), which choose a few thousand of keypoints, infer that it is not the quantity of the correspondences but the quality of the correspondences that matters for accurate localization.

To complement this experiment, we consider a third set of 4800 correspondences (number of correspondences chosen by DSAC++ and DSAC*) and compute the run-time of the pose calculation step, i.e., the PnP in the RANSAC framework. We use the OpenCV python implementation [19]. Comparing set 1 to set 3, the run-time was improved by 14 times for the OldHospital set and 15 times for the KingsCollege set. With a Python implementation using Pytorch [106], one network forward pass to obtain the 2D Keypoints and the 3D coordinates needs around 8ms on a single Nvidia GeForce GTX TITAN X.

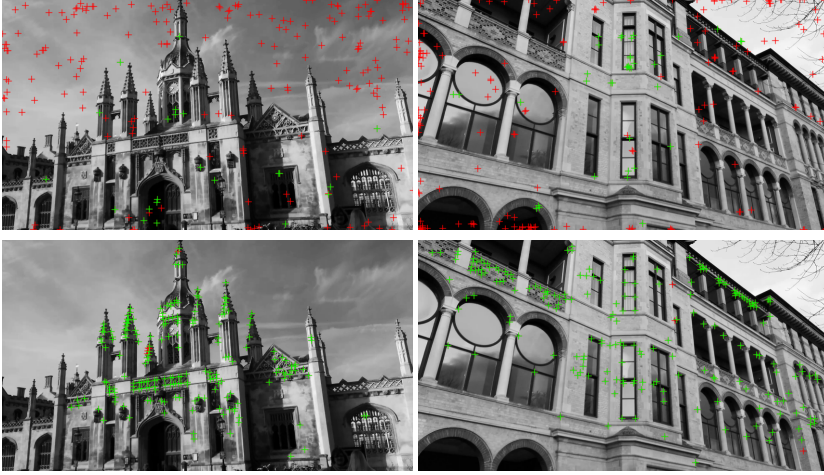


Figure 2.3: Importance of selected Keypoints: The upper images show keypoints that are predicted with lower scores (set 2 of Tab. 2.4). The lower images show keypoints that are predicted with higher scores (set 1 of Tab. 2.4). Green color denotes inliers, the set of keypoints that resulted in the pose hypothesis from RANSAC. Outliers are denoted by red. Left: King’s College, right: Old Hospital.

2.4.7 Observations

We observed by training on the Great Court landmark of Cambridge Dataset [70] that the network could not learn the 3D coordinates. To our understanding, this could be because the 3D scene is far from the camera. Despite this limitation, the network learned to avoid 2D keypoints selection from the grass spaces and the sky, as seen in Fig. 2.4. Similarly, training on the Aachen dataset did not converge. A solution could be to cluster the scene and learn each division separately. From a different perspective, and for general cases where the number of selected correspondences is very low, the threshold can be lowered to select more keypoints and avoid localization failure cases.

2.5 Conclusions

We have presented a camera 6DoF global pose estimation method from a single RGB image that exploits discriminatory image regions to mitigate outliers and localize accurately and efficiently. Our method learns to carefully choose good features and avoid occlusions and other unreliable regions such as sky, streets, trees, bushes, pedestrians, and cars. We show that avoiding these regions minimizes outliers, which allows the selection of a low number of correspondences to estimate a pose, improving localization accuracy and efficiency. To the best of our knowledge, our work is the first to exploit the concept of triangulated features from structure-from-motion method to directly



Figure 2.4: Keypoints selected from the GreatCourt landmark. The grass, the building’s roofs, and the sky are avoided for being non-discriminatory and thus irrelevant for localization.

learn reliable and discriminatory image regions and the first to learn the 3D scene estimation and 2D keypoints detection in one framework. All of this combined has led our work to surpass other learning-based and feature matching/aligning-based methods on scenes from the Cambridge Landmarks dataset in terms of localization accuracy and efficiency. We aim to explore directions for scaling up the localization scope of this work by, for example, pairing the work with an image retrieval system. We aim further to explore the potential of our keypoints detector for matching-based localization approaches by, for example, pairing it with learned or hand-crafted descriptors.

2.6 Further Results

Fig. 2.5 shows more samples of PixSelect keypoints and predicted plain heatmaps from Cambridge landmarks dataset.

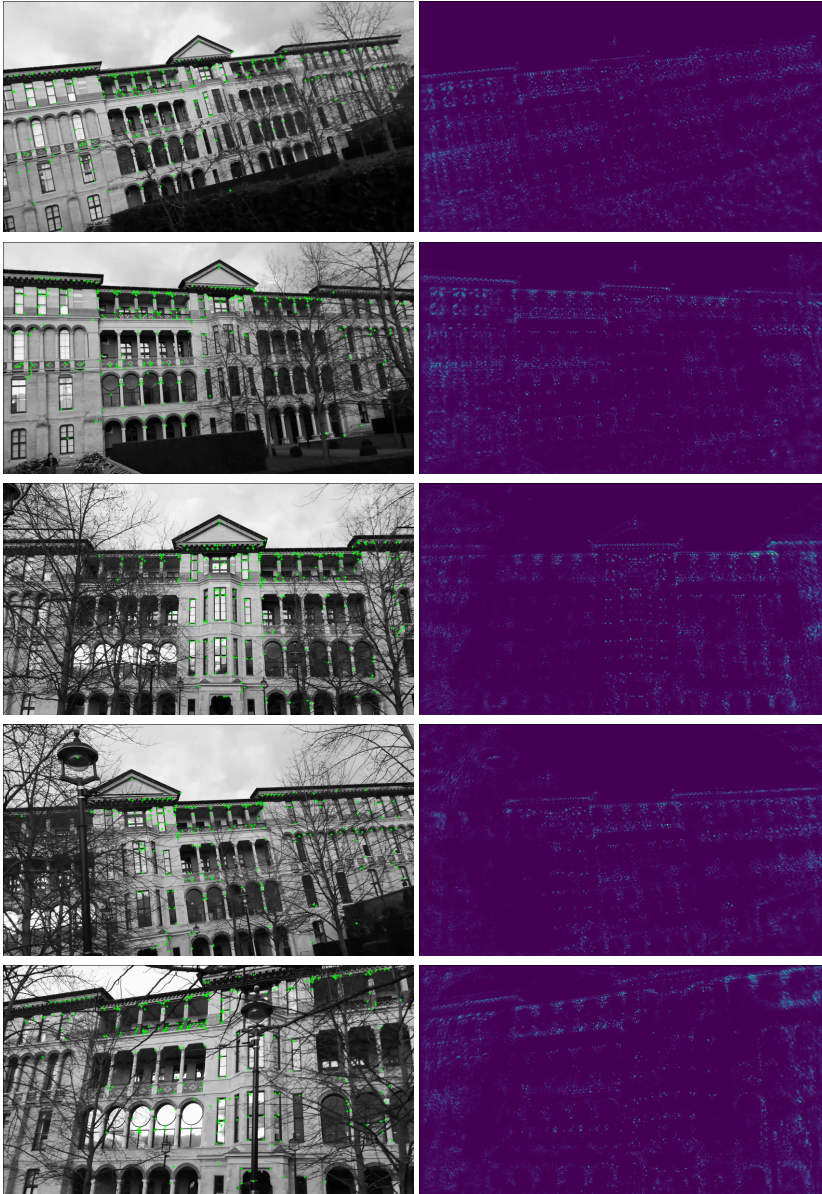




Figure 2.5: PixSelect selects reliable keypoints for localization. It abstains from picking features from dynamic objects like cars and pedestrian, changing anatomies like trees, and non-informative regions like the sky.

Chapter 3

Global Localization: Utilizing Relative Spatio-Temporal Geometric Constraints from Adjacent and Distant Cameras[†]

Re-localizing a camera from a single image in a previously mapped area is vital for many computer vision applications in robotics and augmented/virtual reality. In this work, we address the problem of estimating the 6 DoF camera pose relative to a global frame from a single image. We propose to leverage a novel network of relative spatial and temporal geometric constraints to guide the training of a Deep Network for localization. We employ simultaneously spatial and temporal relative pose constraints that are obtained not only from adjacent camera frames but also from camera frames that are distant in the spatio-temporal space of the scene. We show that our method, through these constraints, is capable of learning to localize when little or very sparse ground-truth 3D coordinates are available. In our experiments, this is less than 1% of available ground-truth data. We evaluate our method on 3 common visual localization datasets and show that it outperforms other direct pose estimation methods.

[†]This chapter is based on the work "Global Localization: Utilizing Relative Spatio-Temporal Geometric Constraints from Adjacent and Distant Cameras," published in the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3358-3365, 2023. The article has been modified to fit the format and requirements of this dissertation.

3.1 Introduction

Estimating the position and orientation of a camera from a single image has been a central research topic in computer vision. It plays a crucial role in many tasks, such as robot navigation and Virtual Reality. With the advent of deep learning in computer vision, recent approaches have leveraged neural networks for data-driven pose estimation. Direct pose regressors [70, 69, 148, 149, 29, 20, 146, 105] use pose labels to learn a direct mapping from image to pose. Although these methods regress pose in real-time, they were shown to be of limited localization accuracy compared to methods that use geometric information to localize [120].

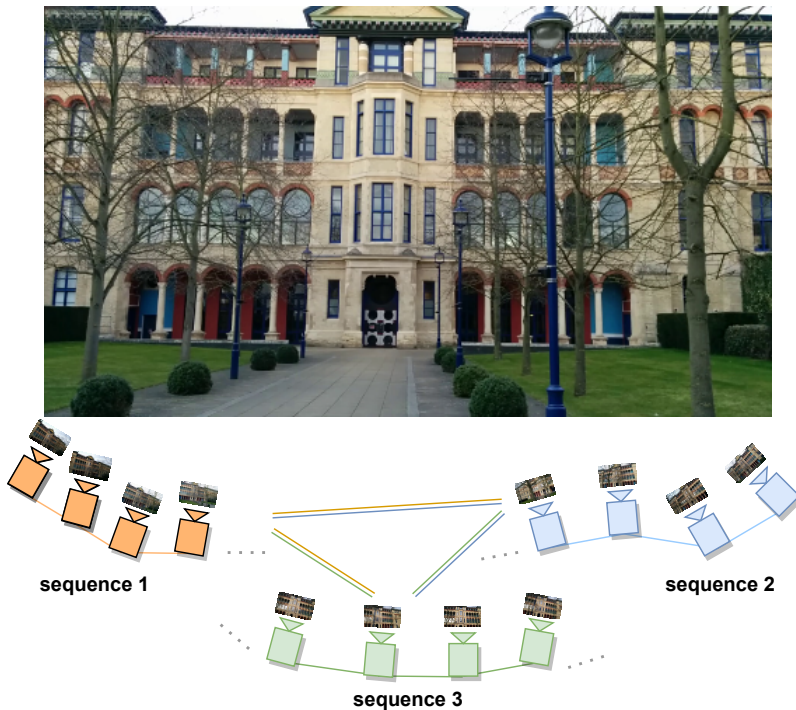


Figure 3.1: Our method utilizes relative geometric information from adjacent cameras as well as distant cameras to optimize the Deep Network weights for the benefit of global localization based on a single image. The proposed work applies relative pose constraints that are obtained from consecutive cameras along each sequence (denoted as single lines between two successive cameras) and from distant cameras across sequences that are far in the spatio-temporal space of the scene (denoted as double lines between two different sequences). These constraints are applied simultaneously for each training iteration.

Most of the direct pose regressors learn from pose targets only. However, these methods ignore that, in most localization scenarios, additional geometric labels, such

as 3D coordinates of the target scene, are available by default. Structure from motion (SfM) is the de-facto method that is used to obtain ground-truth poses on outdoor scenes. SfM obtains camera poses and a 3D point cloud of the scene. In most indoor scenes where classic feature-matching methods may fail because of repetitive structures or aliasing, depth sensors are used to track camera poses and reconstruct the scene. So, by default, obtaining ground-truth poses for training direct pose regressors also delivers geometric scene information. If depth and poses are given, 3D scene coordinates can be obtained by back-projecting the depth to the 3D scene. Vice versa, if 3D scene coordinates are available, depth can be obtained by projecting the 3D coordinates into the camera frames relying on the available poses. As a result, 3D coordinates in a global reference frame, 3D coordinates in the camera frame (obtained from depth), and camera poses relative to the global reference frame are available for training. Direct pose regressors estimate the pose directly by regression without considering this geometric information. However, given the geometric scene information, a Deep Network can be trained to learn the geometry of the scene, which, in return, can be used to localize. We exploit this geometric information in our method. Since the problem at hand is re-localizing a single image (only one image is available at inference time) in a previously mapped area, the geometric information of the scene is needed to obtain an absolute pose in metric units. These can be made available by either saving the 3D scene models of the environment (such as the output of SfM) or obtaining this information from the image while localizing. The latter is favorable, as the former requires a considerable additional memory, which is not anticipated when dealing with the problem of localizing a single image. Besides, utilizing saved ground-truth 3D models at inference time imposes further computations, such as the ones that are driven by feature matching, which is needed to form correspondences between the query image and the 3D model.

In our work, we propose to train a Deep Network, which makes use of the readily available geometric labels, to localize a camera given a single image. At training time, the method employs the available camera poses in a global reference system, the 3D coordinates relative to this global reference system, and the 3D coordinates in the camera reference system to train a Deep Network to learn this geometric information. At inference time, the 3D coordinates in the global frame and those in the camera frame are estimated from a single image. We refer to these as map representations. To further constrain the map representations, we propose to utilize a network of simultaneous spatial and temporal relative geometric constraints. Specifically, we take advantage of multiple sequences of images wherein the sequences are taken at different times and in different scene spatial locations (Fig. 3.1). To fulfill this, we apply losses utilizing the relative poses not only between the successive samples of the same sequence (along the sequence) but also among different samples in different sequences (across sequences). Here, our method is not learning a temporal solution (localize the camera at a time instant given the camera poses/geometry at N previous time instants). Instead, we use the sequences of images to impose more geometric constraints on the training process.

Given the two learned map representations, our method localizes the camera by

aligning them using a classic rigid alignment method, Kabsch [65]. However, localizing from the learned 3D map representations leads to inferior results due to imperfections in the learned maps. This is due to limited ground-truth labels' availability or the presence of unseen objects in query images, such as dynamic objects or occlusions. To account for that, our method estimates weighting factors to guide the rigid alignment.

In summary, our contributions are:

- We propose utilizing a network of relative geometric constraints along and across sequences to benefit global camera localization from a single image. These geometric constraints are computed as relative poses between camera frames that are adjacent or distant in space and time of the scene. Constraints from different sequences are applied simultaneously to update the Deep Network weights in each training iteration.
- Our method learns to localize when little or very sparse ground-truth training 3D coordinates are available. We show that when less than 1% of potential 3D coordinates are available.
- Our method outperforms state-of-the-art direct pose estimation methods.

3.2 Related Works

Many recent works have addressed the camera global pose estimation. These works are categorized based on how the method computes a pose: direct or indirect.

Indirect methods are methods whose pose estimation consists of more than one step, like obtaining many pose hypotheses for further processing/refinement and/or condition on external dependencies like image retrieval. Compared to direct methods, they include additional probabilistic components such as database querying, hypotheses sampling, pose refinement, RANSAC [43], etc. Active Search [123], a classic structure-based method, provides a prioritization scheme to establish matches between the extracted 2D features from a query image and the 3D points from a given 3D model and terminates correspondence search once enough matches are found. The pose is then obtained from the 2D-3D correspondences in a RANSAC framework [43]. Instead of utilizing the 3D model at inference, other methods [16] [18] [3] learn the 3D scene and regress it at inference. Similar to classic methods, a pose is computed from the set of 2D-3D correspondences using a perspective n-point pose solver in a RANSAC scheme [43]. The pose is then refined with the inliers. PixLoc [119], another indirect method, uses an image retrieval method to retrieve the most relevant database images to a query image, in addition to pose priors and a reference 3D SfM model to estimate a pose. These methods obtain accurate poses and exceed direct methods. In our work, the pose estimation is a single-step point cloud alignment.

Direct methods estimate camera pose in a deterministic way using a fixed pipeline without depending on external processes like retrieval from a database of images, match-

ing features, sampling of pose hypotheses, and refinement of selected hypotheses using matched inlier correspondences. Pose regressors like PoseNet [70] and followers fall into this category.

Direct methods estimate pose through regression [70, 69, 148, 149, 29, 20, 146, 105] or a single-step rigid alignment [12]. Pose regression methods learn to map the input image directly into a 6 DoF output pose. A CNN Network learns a function that encodes the input into a latent feature. One or more regression layers map the latent image representation into the pose. These methods followed different learning strategies to encode beneficial information in the latent vector, such as LSTMs [148], 3D model [69], attention [149], graph neural networks [165], and sequence of images [29, 20, 146, 149, 105, 165, 164]. The common practice is that these methods model the geometric pose estimation problem as a regression task. In contrast to that, we estimate two geometric representations of the scene and use them to estimate the pose by aligning the two 3D representations in one step. Some of the methods that utilize a sequence of images encode the temporal information along the same sequence consecutively through recurrent neural networks, which require the sequence of images also at inference [29, 105, 164]. Opposite to that, we use information from sequences only for training and to optimize the two geometric map representations. At inference time, we localize from a single image. Furthermore, none of them utilizes simultaneous spatial and temporal geometric relative information along and across sequences. Another direct method that relies on rigid alignment for pose estimation is StructureAware [12]. However, it learns the map from ground-truth targets only, which may be of limited availability. In contrast to that, our method exploits close and distant spatio-temporal geometric information across the scene. These constraints are obtained from the available ground-truth data without additional sensory inputs.

3.3 Method

3.3.1 Overview

This work addresses the problem of a single image pose estimation, i.e., estimating the pose of a camera with respect to a previously mapped area of a given global reference coordinate system from a single image. Figure 4.2 shows an overview of our method.

During training, we use the target 3D global scene coordinates, the depth, and the ground-truth poses to supervise the training of the Deep Network to estimate two representations of the map as seen by the image. The first is a set of 3D points with coordinates relative to the global coordinate frame of the scene, and the second is a set of 3D coordinates as seen by the camera (relative to the camera frame). For the latter, the Deep Network estimates the depth, which is back-projected, given the camera's intrinsic parameters. The two map representations correspond to each other, i.e., no matching is required to find 3D-3D correspondences. In addition, the Deep Network estimates a corresponding set of weighting factors. To estimate the camera pose, we

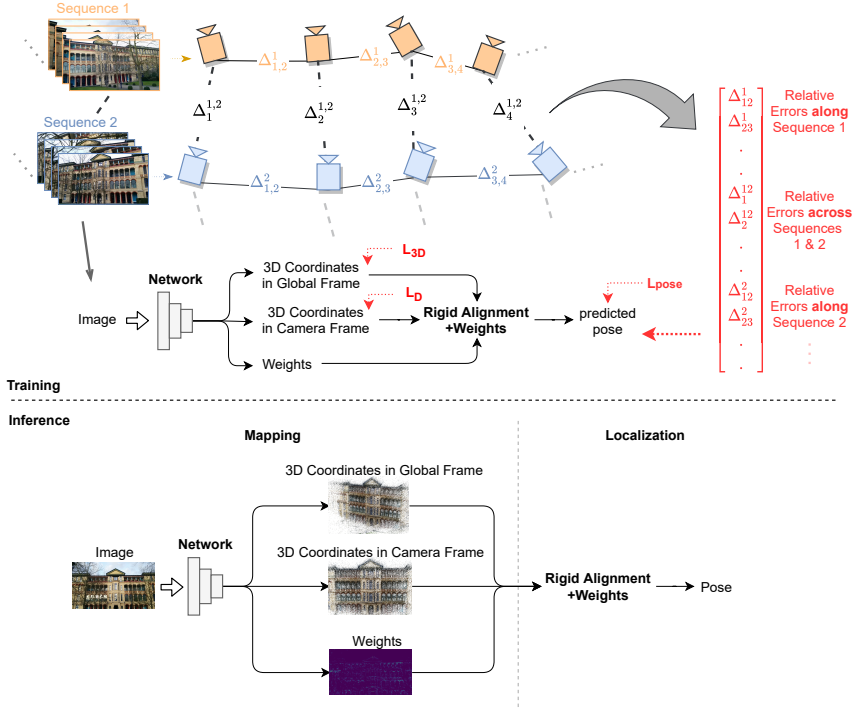


Figure 3.2: A diagram of our method. At training time, sequences of images are used to impose relative geometric constraints. For each image that is fed to the Deep Network, the following is predicted: the corresponding 3D coordinates in the global reference frame (Sec. 3.3.2), the depth which is used to estimate the 3D coordinates in the camera frame (Sec. 3.3.3), and weighting factors (Sec. 3.3.6). Rigid alignment is used to align the two 3D point clouds to estimate a pose. L_{3D} , L_D , and L_{pose} are losses that are applied during the training phase. In addition, relative pose errors along and across sequences are utilized as an additional signal to train the Deep Network (Sec. 3.3.5). $\Delta_{i,j}^k$ stands for the relative pose error between two successive cameras i and j in the same sequence k (along sequences). $\Delta_i^{k,l}$ stands for the relative pose error between the frame i of each of the sequences k and l (across sequences). At inference, an image is fed to the Deep Network to estimate the two map representations. Given that, a pose is obtained through weighted rigid alignment.

use a differentiable Kabsch algorithm [65] to align the two maps according to the estimated weighting factors. The alignment is a single-step closed-form solution. With a differentiable Kabsch algorithm, the whole Deep Network can be trained in an end-to-end manner.

In the following, the components of the proposed method are described in detail. For all the equations below, we denote the predicted counterpart of any ground-truth term with a hat.

3.3.2 3D Points in Global Coordinate Frame

The proposed work estimates the 3D coordinates of the image pixels relative to the global reference system of the scene. This set of 3D coordinates forms the first map representation. The difference between the predicted and the available reference 3D scene coordinates is minimized to train the Deep Network to estimate a global map representation. This is defined as follows:

$$L_{3D} = \frac{1}{M} \sum_i^M \|\hat{\mathbf{g}}_i - \mathbf{g}_i\|_2, \quad (3.1)$$

where $G = \{\mathbf{g}_1, \dots, \mathbf{g}_M\}$ is the set of the ground-truth 3D global coordinates that corresponds to the set of predicted 3D coordinates $\hat{G} = \{\hat{\mathbf{g}}_1, \dots, \hat{\mathbf{g}}_M\}$ and M is the number of considered points for learning. We apply this loss where the 3D coordinates are available. For this purpose, we create a mask that indicates where the 3D coordinates are valid (available) and apply the loss where the mask is valid. For outdoor scenes, small and very sparse 3D coordinates can be available. We show that our method can learn to localize when less than 1% of ground-truth 3D coordinates are available.

3.3.3 3D Points in Camera Coordinate Frame

For learning the second map representation, i.e., the set of 3D coordinates from the perspective of the camera, the available depth labels are used to guide the Deep Network during training. For this goal, the Deep Network is supervised by a loss that minimizes the difference between the predicted and available ground-truth depth as follows:

The depth is learned using L_1 loss:

$$L_D = \frac{1}{M} \sum_i^M |\hat{d}_i - d_i|, \quad (3.2)$$

where \hat{d}_i and d_i are the predicted and ground-truth depth values, respectively. Each of them corresponds to pixel i from the set M of considered pixels. If depth is not directly available, it can be obtained by projecting the 3D coordinates into the image using the ground-truth poses. Since the depth or the 3D global coordinates can be obtained from each other, the mask of availability for both data is the same in most cases.

The depth is then transformed to 3D coordinates in the camera frame using $\hat{\mathbf{c}}_i = \hat{d}_i \mathbf{K}^{-1} \mathbf{u}_i$, where \mathbf{u}_i , \mathbf{K} , \hat{d}_i , and $\hat{\mathbf{c}}_i$ denote the homogeneous pixel coordinates, the camera intrinsic matrix, the depth, and the corresponding point in the camera frame, respectively.

3.3.4 Pose Estimation

We use rigid alignment to align the two corresponding map representations. The two maps are explicitly corresponding to each other. We solve for the pose using the Kabsch algorithm [65]. In the training phase, we utilize its underlying differentiable singular value decomposition (SVD) to pass gradients from a pose loss to update the Deep Network weights. To estimate the predicted translation vector $\hat{\mathbf{t}}$ and rotation matrix $\hat{\mathbf{R}}$, a cost function can be defined as:

$$\arg \min_{\hat{\mathbf{R}}, \hat{\mathbf{t}}} \sum_i^M \|\hat{\mathbf{g}}_i - \hat{\mathbf{R}}\mathbf{c}_i - \hat{\mathbf{t}}\|_2. \quad (3.3)$$

$\hat{\mathbf{R}}$ and $\hat{\mathbf{t}}$ are recovered using a single-step closed-form solution with singular value decomposition, which is parameter-free and differentiable.

Consequently, we define the pose loss as:

$$L_{pose} = L_{tr} + L_{rot}, \quad (3.4)$$

where L_{tr} and L_{rot} stand for the translation and the rotation losses, respectively. We define the translation loss as:

$$L_{tr}(\mathbf{t}_i, \hat{\mathbf{t}}_i) = \|\mathbf{t} - \hat{\mathbf{t}}\|_2, \quad (3.5)$$

where \mathbf{t} and $\hat{\mathbf{t}}$ are the ground-truth and predicted translations, respectively. We find that utilizing L_2 loss for the translation error supports the overall training of our method better than L_1 . With different representations for the rotation, we found that the following definition:

$$L_{rot}(\mathbf{R}_i, \hat{\mathbf{R}}_i) = \cos^{-1} \left(\frac{1}{2} (\text{trace}(\hat{\mathbf{R}}\mathbf{R}^{-1}) - 1) \right) \quad (3.6)$$

works well in combination with the other losses defined in our work. \mathbf{R} is the ground-truth rotation matrix and $\hat{\mathbf{R}}$ stands for its predicted counterpart.

3.3.5 Distant and Adjacent Spatio-Temporal Constraints:

We introduce a novel network of simultaneous relative geometric constraints that cover the spatial and temporal distributions of the scene. The motivation is to constrain the Deep Network to encode better the two map representations for the best pose estimation. For this goal, we apply geometric constraints not only along the consecutive camera frames but also across different sequences that are distant in space and time

(of data collection). These sets of constraints are applied simultaneously. While the ground-truth labels take care of the direct supervision of each frame, these relative geometric constraints introduce further training signals that cover the distant and adjacent spatio-temporal scene geometry. We show in sections 3.4.3 and 3.4.4 the effectiveness of these constraints in learning to localize from little or sparse ground-truth 3D coordinates labels.

For the mentioned purpose, we set these geometric constraints to relative poses between consecutive cameras (close in spatial location and time) and between cameras that are distant in spatial location and time (time of collection of the dataset). We are capable of using relative poses from distant cameras for training because we do not formulate the problem as temporal-based localization (i.e., localize frame i from previous N frames). With this setting, our work localizes from a single image at inference.

For a given image, its predicted pose is computed by aligning the two predicted map representations. The error between a predicted pose and a ground-truth pose guides the training of the Deep Network by passing the gradients via the differentiable singular value decomposition module. A relative pose can be computed between any two cameras located at different locations in the scene. Consequently, we compute relative pose error as a difference between a relative pose computed from two predicted poses and its counterpart computed from the corresponding ground-truth poses. By computing these relative poses between adjacent cameras in a sequence and simultaneously between cameras that are distant, we create a network of geometric constraints at every training iteration. These add additional training signals that further constrain the training.

For the equations below, we denote a pose with a rotation matrix \mathbf{R} and a translation vector \mathbf{t} as \mathbf{T} . The relative pose between two camera frames i and j is defined as:

$${}^i\mathbf{T}_j = \mathbf{T}_i^{-1} \cdot \mathbf{T}_j, \quad (3.7)$$

where \mathbf{T}_i and \mathbf{T}_j are the camera poses in the global frame for images i and j , respectively. Consequently, the relative pose error for two camera frames i and j is defined as:

$$L_{relpose} = L_{tr}({}^i\mathbf{t}_j, {}^i\hat{\mathbf{t}}_j) + L_{rot}({}^i\mathbf{R}_j, {}^i\hat{\mathbf{R}}_j), \quad (3.8)$$

where ${}^i\mathbf{t}_j$ and ${}^i\mathbf{R}_j$ are the translation and rotation parts of the ground-truth relative pose ${}^i\mathbf{T}_j$, respectively. Similarly, ${}^i\hat{\mathbf{t}}_j$ and ${}^i\hat{\mathbf{R}}_j$ are, subsequently, the translation and rotation parts of the predicted relative pose ${}^i\hat{\mathbf{T}}_j$.

For K different sequences, each of N camera frames, the relative pose errors along the sequences are computed according to:

$$L_{Along} = \sum_k^K \sum_i^{N-1} L_{tr}({}_k^i\mathbf{t}_{i+1}, {}_k^i\hat{\mathbf{t}}_{i+1}) + \sum_k^K \sum_i^{N-1} L_{rot}({}_k^i\mathbf{R}_{i+1}, {}_k^i\hat{\mathbf{R}}_{i+1}), \quad (3.9)$$

where the term ${}^i_k \mathbf{t}_{i+1}$ means the translation vector of the relative ground-truth pose between consecutive cameras i and $i + 1$ of sequence k . The same notation applies to the other terms in the equation. Consequently, the relative pose errors across the sequences are computed according to:

$$L_{Across} = \sum_i^N \sum_k^{K-1} L_{tr}({}^k_i \mathbf{t}_{k+1}, {}^k_i \hat{\mathbf{t}}_{k+1}) + \sum_i^N \sum_k^{K-1} L_{rot}({}^k_i \mathbf{R}_{k+1}, {}^k_i \hat{\mathbf{R}}_{k+1}). \quad (3.10)$$

Without confusing the terms of this equation with (3.9) above, the term ${}^k_i \mathbf{t}_{k+1}$ denotes the translation vector of the relative ground-truth pose between two distant cameras with index i , one in sequence k , and the other in sequence $k + 1$. The same notation applies to the other terms in the equation.

3.3.6 Weights

To account for the elements that can degrade the localization performance, we predict a set of weights $W = \{w_i, \dots, w_M\}$ to guide the rigid alignment. These elements can be unseen objects such as dynamic objects, occlusions, irrelevant scene areas such as the sky, or inaccuracies in the training data. These weights measure the contribution of each 3D-3D correspondence to the rigid alignment and are learned during training. For this purpose, we consider a weighted version of the rigid alignment.

In this section, we remove the distinction between predicted quantities (denoted previously by $\hat{\cdot}$) and ground-truth ones. The weighted minimization goal is defined as:

$$\arg \min_{\mathbf{R}, \mathbf{t}} \sum_i^M w_i \|\mathbf{g}_i - \mathbf{R}\mathbf{c}_i - \mathbf{t}\|_2, \quad (3.11)$$

where w_i indicates how much a pair of points contribute to the pose estimation. The algorithm works as follows: the translation \mathbf{t} of the pose is removed by centering both point clouds:

$$\boldsymbol{\mu}_g = \frac{\sum_i w_i \mathbf{g}_i}{\sum_i w_i}, \quad \bar{\mathbf{G}} = \mathbf{G} - \boldsymbol{\mu}_g$$

$$\boldsymbol{\mu}_c = \frac{\sum_i w_i \mathbf{c}_i}{\sum_i w_i}, \quad \bar{\mathbf{C}} = \mathbf{C} - \boldsymbol{\mu}_c.$$

\mathbf{R} and \mathbf{t} are then recovered with SVD as follows:

$$\mathbf{U}\mathbf{S}\mathbf{V}^T = \text{svd}(\bar{\mathbf{C}}^T W \bar{\mathbf{G}})$$

$$s = \det(\mathbf{V}\mathbf{U}^T)$$

$$\mathbf{R} = \mathbf{V} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & s \end{pmatrix} \mathbf{U}^T$$

$$\mathbf{t} = -\mathbf{R}\boldsymbol{\mu}_c + \boldsymbol{\mu}_g$$

Hence, to train the Deep Network to learn these weights, we fill in the predicted rotation and translation of (3.5) and (3.6) by the values obtained from (4.1).

3.4 Experiments

3.4.1 Datasets

We conduct our experiments on three common visual localization datasets: the outdoor Cambridge Landmarks [70] and the indoor datasets: 7Scenes [132] and 12Scenes [147].

Cambridge landmarks is an outdoor relocalization dataset that contains RGB images of six large scenes, each covering a landmark of several hundred or thousand square meters in Cambridge, UK. The provided reference poses are reconstructed from SfM. The authors provide the train and test splits. We use the provided SfM models to obtain the ground-truth 3D points for each image. Depth is obtained by projecting the 3D points into the camera frame using the ground-truth poses. Following previous works [148, 104, 12, 22], we do not conduct experiments on the street and Court landmarks.

7Scenes is a RGB-D indoor relocalization dataset consisting of 7 scenes depicting difficult scenery such as motion blur, reflective surfaces, repeating structures, and texture-less surfaces. Several thousand frames with corresponding ground-truth poses are provided for each scene’s train and test split. To compute the ground-truth 3D global scene coordinates for training, we use the rendered depth maps from [14] as these are registered to the RGB images.

12Scenes is an indoor dataset comprising 12 RGB-D sequences. Compared to 7scenes, it covers larger indoor environments with a smaller number of training images of several hundred frames for each scene. We obtain the 3D global coordinates in the same manner as for 7Scenes by using the rendered depth maps from [13].

3.4.2 Architecture and Setup

We implement the proposal using a single fully convolutional Deep Network with skip connections. The Deep Network takes a RGB input image. After three residual skip

connections, the Deep Network branches out into three branches. Each branch corresponds to one of the predictions. The first is a 3-channel output corresponding to the X , Y , and Z coordinates in the global frame. The second is a one-channel depth prediction. The third is also of one channel that stores the weights for the rigid alignment. We apply stride-2 convolutions to downsample the input resolution by a factor of 8. We add Relu [103] as a non-linear activation after each layer except the last output layers. For the weights prediction, we use Sigmoid as activation for the last layer.

We resize the input images to a standard 480 px height and normalize them by mean and standard deviation. We do not apply any image scaling as augmentation. Thus, the input resolution is kept during the training. For each scene, the ground-truth global 3D coordinates mean is subtracted from the predictions, which is then added at inference. During training, we apply, on the fly, color jittering and random in-plane rotations in the range $[-30^\circ, 30^\circ]$.

We use the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and a weight decay of 5×10^{-4} and a learning rate of 10^{-4} . All losses are applied with equal weighting factors of 1.

For our experiments below, we obtain our network of distant spatio-temporal constraints from 2 sequences, each of 8 consecutive camera frames. The two sequences are sampled randomly for every training iteration. We reshape the input to be of dimension (B, C, H, W) , where B , C , H , and W are the batch size, number of channels, height, and width of the image, respectively, with B being 16 (2×8).

Following other works, we report the pose error in the experiments below as two components: median translation error (in meters) and median rotation error (in degrees).

3.4.3 Learning From Little or Sparse Ground-truth

We show the effectiveness of our method for localization when little or very sparse ground-truth data is available.

We use the backbone described above (section 4.4.2), which downsamples the input resolution by 8. For this resolution, we subsample the 3D ground-truth coordinates by a factor of 8. We apply subsampling instead of interpolation in order to keep the accuracy of the ground-truth. We perform the experiments on the outdoor Cambridge Landmarks, which provide sparse but enough ground-truth 3D coordinates per image. Subsampling the sparse 3D coordinates by a factor of 8 results in a very small set of 3D coordinates. This results in a percentage of ground-truth coordinates below 1% of the possible pixels that could have ground-truth data for the Cambridge landmarks,

We evaluate the localization performance of our method by comparing it to two methods. The first is the direct utilization of the ground-truth data, which we take as a baseline. The second is DSAC* [18], a state-of-the-art visual localization method. To perform a fair comparison, we run the three methods on the same backbone using the

same augmentation and optimizer settings (as described in section 4.4.2).

Baseline: We apply (3.1) to utilize the ground-truth 3D coordinates, where they are available for training. For evaluation, we obtain poses using perspective-n-points algorithm [48] in a RANSAC framework [43] from all the 2D-3D correspondences (Open-CV implementation [19]). The final pose is refined from the inliers. Since, in this case, the weights and the 3D camera coordinates are not relevant, their corresponding branches are kept frozen (non-trainable).

DSAC* [18]: We consider the RGB+3D model version of DSAC*, which requires a 3D model, camera poses, and images. DSAC* learns the 3D scene in two steps. In the first step, it utilizes the 3D ground-truth coordinates to learn the geometry of the scene. In the second step, it implements a fully differentiable pose optimization. This applies a robust fitting of pose parameters using differentiable RANSAC [18]. For the first stage, we take the results from the baseline above. In the second stage, the training from the baseline is further finetuned with the differentiable RANSAC of DSAC*. We use the DSAC* evaluation scripts that sample and refine the best-selected pose hypothesis for evaluation.

Ours: We train the Deep Network using our method by applying our relative pose constraints along (3.9) and across (3.10) sequences simultaneously, the pose loss (4.2) in addition to utilizing the little available ground-truth ((3.1) and (3.2)). For evaluation, we obtain poses by aligning all the correspondences from the two map representations (3D coordinates in global and camera frames) using weighted rigid alignment.

We present the results in Tab. 3.1. For the output resolution of 60×107 , which forms 6420 ground-truth points, each training image frame hosts, on average, less than 1% of that. That is 34 and 15 points out of 6420 per frame on average for college and hospital scenes, respectively.

Table 3.1: Results of the experiment of Sec. 3.4.3. The second row lists the average number of available ground-truth 3D coordinates per image and the corresponding % of the total number of possible ground-truth (GT) points (6420) for output resolution of 60×107 .

	Scenes	Cambridge			
		College	Hospital	Shop	Church
Available GT: count / %		34 / 0.53%	15 / 0.24%	16 / 0.24%	35 / 0.55%
Methods					
Baseline		35.37m, 92.47°	33.98m, 120.12°	9.54m, 109.65°	24.13m, 109.78°
DSAC*		35.79m, 92.69°	35.59m, 157.36°	10.50m, 102.68°	32.15m, 134.74°
Ours		0.52m, 1.39°	0.78m, 3.94°	0.42m, 1.71°	0.94m, 4.55°

Analysis: Our geometric constraints from along and across sequences significantly reduce localization errors when learning from a very small number of 3D ground-truth coordinates. This implies the effectiveness of our proposed work in learning to local-

ize, provided a small number of ground-truth 3D coordinates. The baseline fails to learn map representations from this little ground-truth data. Consequently, applying the fully differentiable pose optimization of DSAC* provides no more support for the training. Failure of DSAC* to adjust the training can be due to different reasons. DSAC* applies different augmentations, mainly scaling of the images, which requires scaling of the target ground-truth data (implemented as an interpolation by DSAC*). In our experiments, we do not apply scaling. Additionally, DSAC* deploys additional training objectives (such as reprojection errors) with multiple hyper-parameters to support the training. In our work, we do not apply reprojection error as a loss. Pairing DSAC* optimization functions as additional constraints for our work forms an interesting application to implement.

The baseline and DSAC* apply RANSAC to sample multiple pose hypotheses and refine the best one. In contrast, our pose estimation is a single direct step. We trained the baseline for a minimum of 600 epochs, while our work obtained the above results after 150 epochs. On GTX Titan X, our Python implementation runs in 20.5ms at inference.

3.4.4 Effectiveness of the Distant Spatio-Temporal Constraints

This experiment shows the effectiveness of employing distant spatio-temporal constraints across sequences on pose estimation. We extend the baseline from the first experiment by including the branch that predicts the depth (consequently, the 3D camera coordinates). So that the baseline can learn the 3D scene, we apply this experiment on the indoor 12Scenes where ground-truth 3D labels are available (dense). The dataset is challenging as a low number of training images are available, but they cover relatively larger areas (compared to 7Scenes).

Baseline + PnP + RANSAC: we train the baseline to learn the two map representations (as our work) from the available dense ground-truth 3D labels (3.1) and (3.2). For localization evaluation, we obtain poses by applying Perspective-n-Points algorithm [48] in a RANSAC framework [43] on all correspondences from the 2D image pixels and the 3D global coordinates. We use OpenCV implementation [19] with 2000 iterations and a reprojection threshold of 10 pixels. The pose is then refined with the inlier correspondences.

Baseline + Rigid: We evaluate the baseline localization by computing the pose using rigid alignment of the two predicted 3D point clouds (3D coordinates in the global frame and 3D coordinates in the camera frame).

Ours + Rigid: We apply our distant spatio-temporal constraints ((3.9) and (3.10)) and the Pose loss (4.2) on the baseline. We compute pose by aligning the two predicted 3D point clouds. Here, the correspondences are weighted equally.

Ours + Rigid + weights: We compute pose by weighted rigid alignment. Here, the correspondences are aligned according to the weights obtained from the weights branch.

The results on the 12 scenes are listed in Tables 3.2 and 3.2.

Table 3.2: The table shows the results of the experiment of Sec. 3.4.4 on the first six scenes of the 12Scenes dataset [147]. The results on the remaining scenes are listed in Tab. 3.3. Errors are reported as median translation error (meters)/median rotation error (degrees). The second row reports, for each scene, the average percentage of available ground-truth (GT) 3D coordinates of the total number of possible ground-truth points per image. The best results are marked in bold.

	Scenes Available GT %	12Scenes					
		Gates362 91%	Manolis 90%	Gates381 92%	Lounge 95%	Kitchen1 90%	Living1 95%
Methods							
Baseline + PnP + RANSAC		0.17/ 2.12	0.10/4.13	0.31/3.85	0.06/2.07	0.11/ 2.27	0.06/1.81
Baseline + Rigid		0.12/4.35	0.23/9.43	0.21/8.61	0.14/4.83	0.09/5.00	0.11/3.46
Ours + Rigid		0.07/2.78	0.10/4.47	0.13/5.06	0.11/3.53	0.07/3.58	0.08/2.60
Ours + Rigid + Weights		0.05/2.19	0.07/2.77	0.08/3.26	0.06/2.10	0.05/2.68	0.06/1.75

Table 3.3: The table shows the results of the experiment of Sec. 3.4.4 on the second half of scenes of the 12Scenes dataset [147]. The results on the first six scenes are listed in Tab. 3.2. The same caption of Tab. 3.2 applies here.

Methods	Scenes Available GT %	12Scenes					
		5a	5b	Bed	Kitchen2	Luke	Living2
		94%	96%	95%	93%	93%	91%
Baseline + PnP + RANSAC		0.25/2.62	0.10/2.51	0.15/2.96	0.04/1.87	0.24/4.16	0.07/2.77
Baseline + Rigid		0.12/5.61	0.17/6.75	0.12/5.95	0.08/3.78	0.19/8.50	0.11/5.72
Ours + Rigid		0.08/3.94	0.13/5.39	0.07/3.48	0.06/2.63	0.10/4.09	0.09/4.04
Ours + Rigid + Weights		0.06/2.23	0.07/2.24	0.04/1.88	0.04/1.91	0.08/2.85	0.06/2.30

Analysis: Utilizing dense ground-truth 3D coordinates helps the baseline to learn the map's representations and localization. By comparing "Baseline + Rigid" to "Baseline + PnP + RANSAC" we observe alternating performances between the two on the different scenes. On some scenes, "PnP + RANSAC" reports lower errors than rigid alignment, while on other scenes, the single-step rigid alignment reports lower errors. "Ours + Rigid" (alignment with equal weightings) shows considerable improvements over "Baseline + Rigid", showing the benefit of our added constraints. "Ours + Rigid" reduces the gap to "Baseline + PnP + RANSAC" on the scenes on which "Baseline + PnP + RANSAC" outperforms "Baseline + Rigid" and expands the gap on the other scenes, on which "Baseline + Rigid" outperforms "Baseline + PnP + RANSAC". Considering the weights for the rigid alignment, we observe additional improvements on all the scenes compared to rigid alignment with equal weighting and improvements on the majority

of the scenes compared to "Baseline + PnP + RANSAC".

Table 3.4: Comparison against State-of-the-art localization methods on the 7scenes [132]. Median translation (m) and rotation (°) errors (lower is better) with the improvements relative to the second best-reported quantity (underlined) are reported.

Methods	7Scenes						
	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
Indirect Methods							
ActiveSearch [123]	0.03/0.87	0.02/1.01	0.01/0.82	0.04/1.15	0.07/1.69	0.05/1.72	0.04/1.01
DSAC* [18]	0.02/1.10	0.02/1.24	0.01/1.82	0.03/0.8	0.04/1.34	0.04/1.68	0.03/1.16
PixLoc [119]	0.02/0.8	0.02/0.73	0.01/0.82	0.03/0.82	0.04/1.21	0.03/1.20	0.05/1.30
Direct Methods							Average
PoseNetGeo [69]	0.13/4.48	0.27/11.3	0.17/13.0	0.19/5.55	0.26/4.75	0.23/5.35	0.35/12.4
PoseNetLSTM [148]	0.24/5.77	0.34/11.9	0.21/13.7	0.30/8.08	0.33/7.00	0.37/8.83	0.40/13.7
Hourglass PN [95]	0.15/6.17	0.27/10.8	0.19/11.6	0.21/8.48	0.25/7.01	0.27/10.2	0.29/12.5
BranchNet [159]	0.18/5.17	0.34/8.99	0.20/14.2	0.30/7.05	0.27/5.10	0.33/7.40	0.38/10.3
GPoseNet [22]	0.20/7.11	0.38/12.3/	0.21/13.8/	0.28/8.83	0.37/6.94	0.35/8.15	0.37/12.5
MapNet [20]	0.08/3.25	0.27/11.7	0.18/13.3	0.17/5.15	0.22/4.02	0.23/4.93	0.30/12.1
AtLoc [149]	0.10/4.07	0.25/11.4	0.16/11.8	0.17/5.34	0.21/4.37	0.23/5.42	0.26/10.5
StructAware [12]	0.08/2.17	0.21/6.14	0.13/7.93	0.11/2.65	0.14/3.34	0.12/2.75	0.29/6.88
GNNPose [165]	0.08/2.82	0.26/8.94	0.17/11.41	0.18/5.08	0.15/2.77	0.25/4.48	0.23/8.78
AtLoc+ [149]	0.10/3.18	0.26/10.8	0.14/11.4	0.17/5.16	0.20/3.94	0.16/4.90	0.29/10.2
MapNet++ [20]	0.10/3.17	0.20/9.04	0.13/11.1	0.18/5.38	0.19/3.92	0.20/5.01	0.30/13.4
SeqEnhance [164]	0.09/3.28	0.26/10.92	0.17/12.7	0.18/5.45	0.20/3.66	0.23/4.92	0.23/11.3
Ours	0.05/1.52	0.15/3.94	0.08/6.24	0.11/2.40	0.11/2.52	0.11/2.50	0.20/5.56
Improvements:							20%/22%

3.4.5 Comparison Against State-of-the-Art Methods

We compare our method against other global pose estimation methods. We list the localization results on the indoor 7Scenes dataset [132] and the outdoor Cambridge Landmarks [70] in tables 3.4 and 3.5, respectively. We compare our results mainly against the direct methods since our method is direct. For completeness, we list the results of the indirect state-of-the-art methods. Looking at the indoor and outdoor results, we observe that our work reports the lowest localization errors on all indoor scenes and the lowest translation errors among the direct methods on the outdoor scenes.

Limitations: Though our work learns the 3D scene as some indirect methods, it reports relatively higher errors. This can be reverted to a limitation in the rigid alignment's ability to guide the learning of the weights. Indirect methods use a more robust outlier filtering approach.

Table 3.5: Localization errors on the Cambridge Landmarks [70]. Median translation (m) and rotation (°) errors (lower is better) with the improvements relative to the second best-reported quantity (underlined) are reported.

Cambridge					
Methods	College	Hospital	Shop	Church	
Indirect Methods					
ActiveSearch [123]	0.13/0.22	0.20/0.36	0.04/0.21	0.08/0.25	
DSAC* [18]	0.15/0.3	0.21/0.4	0.05/0.3	0.13/0.4	
PixLoc [119]	0.14/0.24	0.16/0.32	0.05/0.23	0.10/0.34	
PixSelect [3]	0.14/0.34	0.14/0.5	0.06/0.50	0.09/0.46	
Direct Methods					Average
PoseNet [70]	1.92/5.40	2.31/5.38	1.46/8.08	2.65/8.08	2.08/6.83
PoseNetGeo [69]	0.88/1.04	3.20/3.29	0.88/3.78	1.57/3.32	1.63/2.86
PoseNetLSTM [148]	0.99/3.65	1.51/4.29	1.18/7.44	1.52/6.68	1.30/5.51
SVS-Pose [104]	1.06/2.81	1.50/4.03	0.63/5.73	2.11/8.11	1.32/5.17
GPoseNet [22]	1.61/2.29	2.62/3.89	1.14/5.73	2.93/6.46	2.08/4.59
MapNet [20]	1.07/1.89	1.94/3.91	1.49/4.22	2.00/4.53	1.62/3.64
StructAware [12]	1.19/2.16	1.11/1.92	0.95/6.82	1.37/4.45	1.16/3.84
GNNPose [165]	0.59/0.65	1.88/2.78	0.50/2.87	1.90/3.29	1.22/2.40
Ours	0.52/1.39	0.78/3.94	0.42/1.71	0.94/4.55	0.67/2.90
Improvements:					42%/-

3.5 Conclusions

This work addresses 6 DoF camera re-localization in a previously mapped area based on a single image. We propose to utilize relative geometric spatio-temporal constraints for training. These constraints are defined as relative poses and are obtained not only from adjacent and consecutive cameras but also from cameras that are distant in the spatio-temporal space of the scene. We apply these constraints simultaneously in every training iteration, and we show their benefit in learning to localize from as little as below 1% of 3D ground-truth coordinates. Our method outperforms other direct methods. A potential direction for further works would be pairing our network of distant and adjacent geometric constraints with a differentiable RANSAC learning scheme and guided by a reprojection error objective function.

Chapter 4

Implicit Learning of Scene Geometry from Poses for Global Localization[†]

Global visual localization estimates the absolute pose of a camera using a single image in a previously mapped area. Obtaining the pose from a single image enables many robotics and augmented/virtual reality applications. Inspired by the latest advances in deep learning, many existing approaches directly learn and regress 6 DoF pose from an input image. However, these methods do not fully utilize the underlying scene geometry for pose regression. The challenge in monocular relocalization is the minimal availability of supervised training data, which is just the corresponding 6 DoF poses of the images. In this paper, we propose to utilize these minimal available labels (i.e., poses) to learn the underlying 3D geometry of the scene and use the geometry to estimate the 6 DoF camera pose. We present a learning method that uses these pose labels and rigid alignment to learn two 3D geometric representations (X, Y, Z coordinates) of the scene, one in the camera coordinate frame and the other in the global coordinate frame. Given a single image, it estimates these two 3D scene representations, which are then aligned to estimate a pose that matches the pose label. This formulation allows for the active inclusion of additional learning constraints to minimize 3D alignment errors between the two 3D scene representations and 2D re-projection errors between the 3D global scene representation and 2D image pixels, resulting in improved localization accuracy. During inference, our model estimates the 3D scene geometry in camera and global frames and aligns them rigidly to obtain pose in real-time. We evaluate our work on three common visual localization datasets, conduct ablation studies, and show that our method exceeds state-of-the-art regression methods' pose accuracy

[†]This chapter is based on the work "Implicit Learning of Scene Geometry From Poses for Global Localization," published in IEEE Robotics and Automation Letters (RA-L), vol. 9, no. 2, pp. 955-962, Feb. 2024. The article has been modified to fit the format and requirements of this dissertation.

on all datasets.

4.1 Introduction

Global camera re-localization has driven many computer vision applications in augmented/virtual reality and robotics. The problem definition, which is to obtain a camera's position and orientation (in metric units) from a single image in a previously mapped area, requires the availability of ground-truth poses for training. With training data that is composed of images and the corresponding poses, these methods follow a common learning process that maps the input image to 6 DoF directly in an end-to-end manner. This learning process goes as follows: the input image is passed to a network that encodes it into a latent vector. Through one or more regression layers, the latent vector is then mapped into a pose as two quantities, one for the translation and the other for the rotation. A pose loss is deployed to update the network weights during training. With a new input image at localization time, the network utilizes its memory formed by the learned weights to estimate the pose. Recent works [70, 69, 22, 148, 104, 95, 159, 20, 149, 105, 29, 146, 81, 165, 128, 98] build upon this scheme with different variations to constrain the weights to obtain a better pose estimate. This approach is attractive for several reasons. Firstly, it provides a pose directly in a single regression step. Secondly, the mapping from image to pose is fast; no classic feature matching is required, making it suitable for real-time applications. Thirdly, the whole pose estimation pipeline is saved compactly as network weights.

However, one limitation of this formulation is that it treats pose estimation as a regression problem, and consequently, it strips out the geometry of the scene from pose estimation. This conditions the design of the deep network's last layer to be fully connected layers. This precludes incorporating geometric constraints such as depth or 3D scene coordinates. These can only be included in the loss terms in the training phase. Otherwise, they are explicitly required during inference (for example, 3D models from structure from motion).

In this work, we regard these limitations and propose a novel approach for global camera re-localization. Similar to other pose regression methods, our method is subject to the constraint of using minimalistic training data: a set of images with their intrinsics and the corresponding poses in a global reference frame. However, our pipeline does not estimate a pose by regression. Instead, it obtains geometric information of the scene, which can be used, in turn, to estimate the pose geometrically from pose labels only. The proposed pipeline learns 3D geometric representations (X , Y , Z coordinates) of the scene as seen by the image, given guidance from ground-truth poses alone, as shown in Fig. 4.1. It takes a single image and obtains 3D point cloud of the scene in two coordinate systems: camera frame and a global reference frame. To accomplish this, we utilize rigid alignment as a means to adjust the network weights in order to ob-

tain two geometric maps. The rigid alignment module aligns the two clouds to obtain a pose. This pose is adjusted through gradient descent while training so as to match the ground-truth pose, thus implicitly adjusting the two geometric representations (3D clouds) as well.

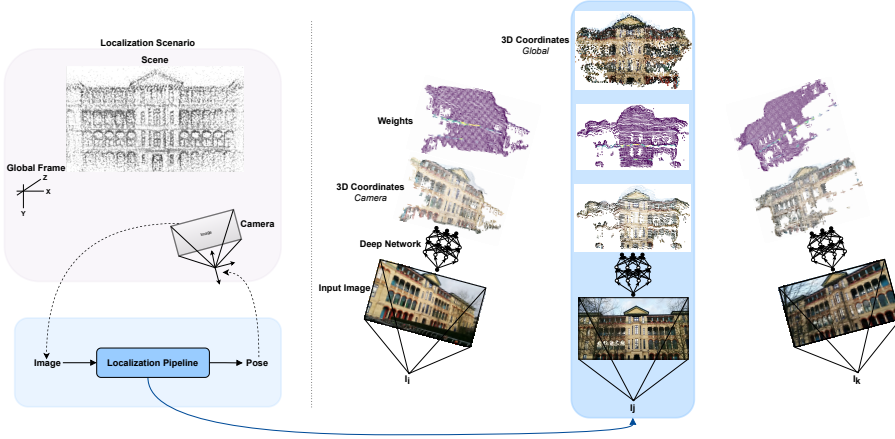


Figure 4.1: Illustration of our visual localization proposal on samples from Cambridge Landmarks (Hospital scene). Our method requires a set of images and the corresponding poses as the only labels for training. Left side: Given a single image, our method estimates the global pose of the camera in a given scene. Right side: we display the intermediate outputs of our proposal, which are used to estimate the pose. For an input image, the proposed pipeline estimates two point clouds and a set of weights. The first point cloud represents the scene geometry (X, Y, Z coordinates) in the camera coordinate frame, while the second point cloud represents the scene geometry in the global coordinate frame. These two point clouds and the predicted weights are used to estimate the camera’s global pose. On the right side, we visualize three sample input images, their corresponding indirectly estimated 3D scene representations (point clouds), and the weights. At the top, on the right side of the figure, we can see only one 3D point cloud, which corresponds to three overlaid point clouds in the global coordinate frame, also estimated by our algorithm for the considered sample images. Though our method implicitly estimates 3D point clouds of the scene in local and global reference frames, it is not a mapping or 3D reconstruction algorithm but a localization algorithm that implicitly learns and uses 3D scene geometry.

In our end-to-end pipeline, deep learning is used to learn scene-specific geometric representations while we perform pose estimation in a geometric manner by parameter-free rigid alignment. It obtains a pose during inference by rigidly aligning the two predicted 3D geometric representations. In contrast to pose regression, the proposed formulation allows the explicit use of additional constraints during training. We minimize re-projection errors in order to limit the deviation of the 3D map representation in the global frame from the corresponding 2D pixels. We complement that by constraining the deviation of the two 3D clouds according to ground-truth pose, which we refer to as consistency loss. This formulation results in higher localization accuracy than state-

of-the-art regression methods. Additionally, we observe that our method can improve both position and orientation localization when finetuned with absolute position labels only. This is useful in applications where initially only a small set of training poses are available, but during operation, more data as partial geometric labels (position information from GPS) is available.

In summary, our contributions are:

- We propose to utilize poses to train a network to learn geometric representations of a given scene implicitly. These are 3D coordinates in camera and a scene/global reference frame. For this goal, we utilize parameter-free and differentiable rigid alignment supervised by pose loss to guide the learning of scene geometry.
- We also propose to employ additional loss terms, specifically, re-projection loss, to constrain the learning of the 3D scene coordinates and introduce a consistency loss term that harmonizes the implicit geometric representations according to the pose.
- Apart from extensive evaluation on public datasets, we conduct ablation studies to evaluate the influence of the three different losses on our method's performance. The ablation studies show that our method can be finetuned using partial pose labels, i.e., with position information alone, and still offers a fairly good performance by improving both position and orientation accuracy.

4.2 Related Work

The problem at hand is monocular global re-localization, which is to obtain metric poses (in meters and degrees) in a previously mapped area. Existing works utilize different sets of labels to train a localization pipeline. While some works utilize 3D geometric information such as 3D scene coordinates and depth, others utilize only pose labels. Similar to pose regression methods, our proposed method learns from pose labels only).

Initial works [70, 69, 22, 148, 104, 95, 159, 20, 149, 105, 29, 146, 81, 165, 128, 98] followed the regression approach and implemented different network architectures, learning strategies, and constraints on the learning process to reduce localization errors. In the following, we briefly review these methods and point out the coincidences and differences with respect to ours. PoseNetLSTM [148] utilizes LSTMs to reduce the dimensionality of the latent vector (that encodes the image) as a way to mitigate overfitting and obtain a more accurate pose estimate. PoseNetLearned [69] improves localization accuracy of their initial work PoseNet [70] by addressing the issue of loss imbalance between the orientation and translation losses by learning weighting factors for these terms using homoscedastic uncertainty. PoseNetGeo [69] further improves the accuracy for some scenes by utilizing explicit 3D coordinate labels through re-projection

loss in addition to pose loss. Though explicit 3D coordinate labels are used, the re-projection loss plays a limited role as it is used to constrain the image latent representation for regression instead of learning 3D scene geometry for pose estimation. Our proposal does not use 3D coordinate labels for training but still learns 3D geometry and uses it for pose estimation. Hourglass PN [95] replaces the initial architecture of PoseNet [70] by a Resnet34 [57] and complements it with up-convolutions to preserve the fine-grained information of the input image, forming an Hourglass-like architecture. BranchNet [159] also adapts the architecture of PoseNet [70] to account for the complex coupling between position and orientation. The split in network branches for position and orientation is performed at an earlier stage of the series of convolutions. Other works [29, 20, 146, 105] utilize sequences of images to gain additional sources of training signals by imposing relative pose constraints that are obtained between consecutive camera frames.

AtLoc [149] proposes a method based on attention to guide the network to output latent image representation that encodes robust objects and features for pose regression. It further utilizes a sequence of images to learn temporally consistent and informative features. CoordiNet [98] embeds pixel coordinates into the convolution operation to encode geometric feature locations into pose regression. It appends two additional channels that contain 2D pixel locations to the input tensor before applying the convolution. MsTransformer [128] proposes a transformer-based approach for localization. It utilizes image latent representations that are obtained from CNN for processing by separate Transformers to regress position and orientation. Transformer encoders are used to aggregate activation maps with self-attention, and decoders convert latent features and scene encoding into pose predictions. With the advances in graph neural networks (GNN), PoGo-Net [81] and GNNPose[165] formulate the pose regression based on GNNs, naturally propagating information between different views for the benefit of pose regression.

Similar to previous regression works, we train from the available image-pose labels for the given datasets. In contrast, we propose using deep learning to implicitly learn representations of the 3D geometry of the scene instead of directly learning the pose regression. Accordingly, we solve for the pose by a single-step closed-form solution through the rigid alignment of 3D scene representations. There are other works that use additional sensing modalities such as LiDAR or labeled depth data or structure from motion results for training and/or inference [18, 3, 12, 4]. However, in this work, we solely focus on training with posed images.

4.3 Method

4.3.1 Overview

Figure 4.2 shows an overview of the proposed method. We propose to use the global camera pose \mathbf{T} of a given input image \mathbf{I} as a label to guide the training of a deep neural

network to obtain representations of the scene.

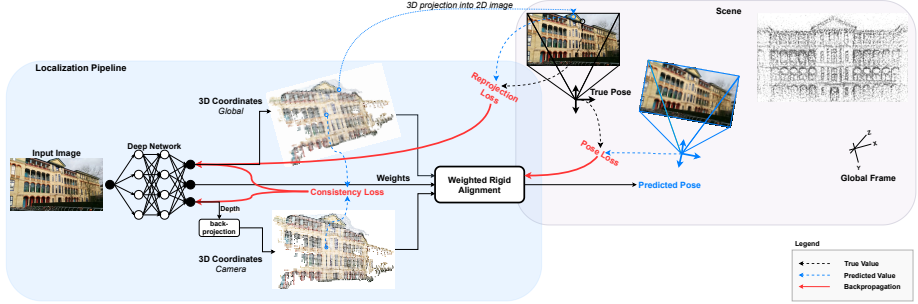


Figure 4.2: A diagram of the proposed training method. Given images and the corresponding global pose labels, the network learns to indirectly (i.e., without explicit labels) estimate two 3D point clouds (one in global and the other in camera coordinate systems) and a set of weights. For the 3D coordinates in the camera coordinate system, the network predicts the depth, which is then back-projected to 3D space using Equation. (4.6). A rigid alignment module aligns the two point clouds according to the weights to estimate pose. The loss terms employed for training are visualized in red.

For that purpose, we define our localization pipeline to take a given image as input and yield two sets of 3D points, each in a different coordinate system. The first one is a set of 3D coordinates $G = \{\hat{\mathbf{g}}_1, \dots, \hat{\mathbf{g}}_M\}$ in the global reference frame of the scene. These are predicted directly by the network. The second one is a set of 3D coordinates $C = \{\hat{\mathbf{c}}_1, \dots, \hat{\mathbf{c}}_M\}$ in the camera frame. For the latter, the network predicts depth, which is then back-projected using intrinsic parameters to get 3D coordinates in the camera frame. Inherently, the two 3D points clouds are matched via the image pixel coordinates.

Using rigid alignment, a pose $\hat{\mathbf{T}}$ can be estimated by aligning the two point clouds. We utilize the Kabsch algorithm [65] for this goal. It is differentiable, parameter-free, and obtains a closed-form solution in a single step. This makes the pipeline end-to-end trainable.

To account for imperfections in predictions, the network predicts a set of weights $W = \{w_1, \dots, w_M\}$, which evaluates how much each 3D correspondence between point clouds from camera and global coordinate frame contributes to the rigid alignment. Given such correspondences, the weighted Kabsch algorithm [65] is then applied to estimate the relative pose from camera coordinate system to the global coordinate system. Given M 3D coordinates, this weighted minimization goal is defined as:

$$\arg \min_{\mathbf{R}, \hat{\mathbf{t}}} \sum_i^M w_i \|\hat{\mathbf{g}}_i - \mathbf{R}\hat{\mathbf{c}}_i - \hat{\mathbf{t}}\|_2, \quad (4.1)$$

which can be described as follows: the translation $\hat{\mathbf{t}}$ of the pose is removed by centering both point clouds:

$$\boldsymbol{\mu}_g = \frac{\sum_i w_i \hat{\mathbf{g}}_i}{\sum_i w_i}, \quad \bar{\mathbf{G}} = \mathbf{G} - \boldsymbol{\mu}_g$$

$$\boldsymbol{\mu}_c = \frac{\sum_i w_i \hat{\mathbf{c}}_i}{\sum_i w_i}, \quad \bar{\mathbf{C}} = \mathbf{C} - \boldsymbol{\mu}_c.$$

Rotation $\hat{\mathbf{R}}$ and translation $\hat{\mathbf{t}}$ are then recovered with SVD as follows:

$$\mathbf{U}\mathbf{S}\mathbf{V}^T = \text{svd}(\bar{\mathbf{C}}^T W \bar{\mathbf{G}})$$

$$s = \det(\mathbf{V}\mathbf{U}^T)$$

$$\hat{\mathbf{R}} = \mathbf{V} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & s \end{pmatrix} \mathbf{U}^T$$

$$\hat{\mathbf{t}} = -\hat{\mathbf{R}}\boldsymbol{\mu}_c + \boldsymbol{\mu}_g.$$

We apply a pose loss to guide the rigid alignment so that the network learns the 3D geometric representations. Given a ground-truth pose \mathbf{T} with rotation \mathbf{R} and translation \mathbf{t} components, a cost function can be defined to minimize the difference between the estimated and ground-truth components. We define the loss as the summation of position loss and the rotation loss:

$$L_{pose} = L_{position} + L_{rotation}, \quad (4.2)$$

where

$$L_{position} = \|\mathbf{t} - \hat{\mathbf{t}}\|_2, \quad (4.3)$$

defines the position error between the computed translation $\hat{\mathbf{t}}$ and the actual translation \mathbf{t} and

$$L_{rotation} = \cos^{-1}\left(\frac{1}{2}(\text{trace}(\hat{\mathbf{R}}\mathbf{R}^{-1}) - 1)\right) \quad (4.4)$$

measures the angular error between computed rotation $\hat{\mathbf{R}}$ and the ground-truth rotation \mathbf{R} .

The predicted pose is adjusted by gradient descent that is guided, during training, by the pose loss equation (4.2), to match the ground-truth pose. This process indirectly adjusts the two geometric representations by passing gradients through the differentiable rigid alignment. The proposed formulation allows for the inclusion of additional constraints that actively guide the optimization of the implicit 3D geometric representations from the poses. Consequently, we introduce a consistency loss to constrain the

geometric predictions to be aligned according to the ground-truth pose. We first transform the 3D points C from camera coordinate frame to global coordinate frame using the ground-truth pose. The consistency loss measures the error between the 3D points G in global coordinate frame and the 3D points C transformed from camera coordinate frame, using the ground-truth poses. We define the consistency loss as:

$$L_{consistency} = \frac{1}{M} \sum_i^M \|\hat{\mathbf{g}}_i - \mathbf{T}\hat{\mathbf{c}}_i\|_2, \quad (4.5)$$

Rather than predicting the 3D coordinates directly, we can adjust the network to predict one quantity, depth. Given the depth, which forms the Z ordinate in the camera perspective, the X and Y are obtained directly from the image pixels and depth, given camera intrinsics parameters. Accordingly, the 3D points C in camera coordinate frame are obtained by back-projecting the depth according to:

$$\hat{\mathbf{c}}_i = \hat{d}_i \mathbf{K}^{-1} \mathbf{u}_i, \quad (4.6)$$

where \mathbf{u}_i , \mathbf{K} , \hat{d}_i , and $\hat{\mathbf{c}}_i$ denote the homogeneous pixel coordinates, the camera intrinsic matrix, the depth, and the corresponding point in the camera frame, respectively.

In addition, the 3D global coordinates are further constrained by utilizing a re-projection loss to minimize the error between the re-projection of the 3D global coordinates into the image frame and the 2D image pixels. It is defined as:

$$L_{reprojection} = \frac{1}{M} \sum_i^M \|\mathbf{u}_i - \pi(\mathbf{T}\hat{\mathbf{g}}_i)\|_2, \quad (4.7)$$

where π projects points from the 3D global frame into the image frame.

With pose labels and the defined formulation, our method implicitly learns geometric representations of the scene. Given an image at inference, the proposed method estimates the scene's geometry and utilizes it for pose computation.

The overall loss is then the weighted combination of the pose loss, the re-projection loss, and the consistency loss:

$$L_{total} = \lambda_p L_{pose} + \lambda_c L_{consistency} + \lambda_r L_{reprojection}, \quad (4.8)$$

where λ_p , λ_c , and λ_r are the losses weighting factors.

4.4 Results

4.4.1 Datasets

We conduct our experiments on three standard visual localization datasets. These are the outdoor Cambridge Landmarks [70], the indoor 7Scenes [132], and the indoor 12scenes [147] datasets. They exhibit different characteristics:

Cambridge landmarks is an outdoor re-localization dataset that covers different landmarks of several hundred or thousand square meters in Cambridge, UK. The provided reference poses are reconstructed from structure from motion. This dataset's challenges arise from a variety of illumination changes due to weather and dynamic objects such as cars and pedestrians. In addition, the training set size is relatively small (a few hundred images).

7Scenes contains seven scenes that depict difficult scenarios, such as motion blur, reflective surfaces, repeating structures, and texture-less surfaces. For each scene's train and test splits, several thousand frames with corresponding ground-truth poses are provided.

12Scenes consists of 12 sequences with challenges similar to 7Scenes dataset. However, compared to 7scenes, it covers larger indoor environments with a smaller number of training images, about several hundred frames for each scene.

4.4.2 Setup

We resize the input images to a standard 480 px height and normalize them by mean and standard deviation. During training, we apply, on the fly, color jittering and random in-plane rotations in the range $[-30^\circ, 30^\circ]$.

We adjust every backbone network used in our experiments to obtain three outputs. The first is a 3-channel output that corresponds to the X , Y , Z coordinates in the global frame. The second is a one-channel depth prediction. Depth is obtained from a Sigmoid function and then scaled to a range of $[0.1 \ 10]$ for indoor and $[0.1 \ 600]$ for outdoor scenes. These hyperparameters are generalizable and adjustable to specific scenes. The third is also of one channel, followed by a Sigmoid, which stores the weights that weigh the contribution of the correspondences to the rigid alignment.

For updating the network weights, we use Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and a weight decay of 5×10^{-4} . We train the whole architecture from scratch for 400 epochs with learning rate 10^{-4} . We set the weighting factors λ_p , λ_c , and λ_r of Equation. (4.8) to 1, 1, and 0.001, respectively, to provide a balanced contribution to gradient updates. The low factor for the reprojection error aims to stabilize the training.

Following previous methods, we report localization errors as median translation (in meters) and median orientation (in degrees) errors for all the experiments below. For some experiments, we also list the average localization errors computed on all scenes of the corresponding datasets.

4.4.3 Results: Comparison to Previous Methods

In this section, we compare our proposed method against the state-of-the-art. We consider the methods that utilize pose labels to train a deep network for global localization

from a single image without including additional supervision signals or more sensory data. To marginalize improvements that may come from using a recent backbone, we implement our method using ResNet34 backbone [57], which previous methods adopt. We report the median localization errors in Table 4.2.

Table 4.1: Comparison against state-of-the-art localization methods on Cambridge Landmarks [70]. The first and second best results are marked in **bold** and underline, respectively. ‘_’ denotes unavailable results.

Method	Backbone	Cambridge				
		College	Hospital	Shop	Church	Average
PoseNetGeo [69]	GoogLeNet	0.88/1.04	3.20/3.29	0.88/3.78	1.57/3.32	1.63/2.86
PoseNetLSTM [148]	GoogLeNet	0.99/3.65	1.51/4.29	1.18/7.44	1.52/6.68	1.30/5.51
GPoseNet [22]	GoogLeNet	1.61/2.29	2.62/3.89	1.14/5.73	2.93/6.46	2.08/4.59
BranchNet [159]	GoogLeNet	—	—	—	—	—
SVS-Pose [104]	VGGNet	1.06/2.81	1.50/4.03	0.63/5.73	2.11/8.11	1.32/5.17
Hourglass PN [95]	ResNet34	—	—	—	—	—
MapNet [20]	ResNet34	0.94/1.99	2.03/3.60	0.80/6.34	1.66/4.01	1.36/3.99
AtLoc [149]	ResNet34	—	—	—	—	—
CoordiNet [98]	ResNet34	0.80/1.22	1.43/2.86	0.73/4.69	1.32/4.10	1.07/3.22
CoordiNet [98]	EffNet b3	0.70/0.92	0.97/2.08	0.69/3.74	1.32/3.56	0.92/2.58
PoGO-Net [81]	Not Applicable	-/0.94	-/1.69	-/2.40	-/2.12	-/1.78
MsTransformer [128]	EfficientNetB0	0.83/1.47	1.81/2.39	0.86/3.07	1.62/3.99	1.28/2.73
GNNPose [165]	ResNet34	0.59/ 0.65	1.88/2.78	0.50/2.87	1.90/3.29	1.22/2.40
Ours	ResNet34	0.48/0.84	0.67/ <u>1.14</u>	0.47/ <u>1.91</u>	0.90/3.05	0.63/ <u>1.74</u>
	MobileNetV3	0.46/0.81	0.61/1.09	0.44/1.71	0.87/2.88	0.60/1.62

As listed, our method, with both backbones, obtains the lowest localization errors on all of the scenes, except for the rotation measurements on the college and church scenes. Even though our method does not rely on labeled 3D ground-truth coordinates, it surpasses PoseNetGeo [69], which uses explicit 3D coordinates. PoseNetGeo [69] obtains pose by regression, which doesn’t directly incorporate this available geometric information. In contrast, our method uses poses to infer the geometry of the scene, which is directly embodied for pose estimation by rigid alignment. AtLoc [149] implements attention to utilize informative regions of a given image for pose regression. On the contrary, our method obtains weighting factors that are directly used to minimize contributions from outliers, thus improving localization accuracy. We show the benefit of these weights in section 4.4.5. Previous methods also implement other strategies such as graph-neural-networks [165, 81], transformers [128] relative poses supervision [20], and LSTMs [148] to better encode the image representation for the task of pose estimation. However, their performances are leveled by regression incapability to utilize geometric quantities directly for localization. Our method uses pose labels to guide the network in learning certain geometric features, which, in return, are used to compute a pose. It lets the network, through pose targets, freely choose the suitable geometric features that are best for localization. Besides using ResNet34 [57] as a backbone, we implement our method using MobileNetV3 [59] due to its efficiency. As shown in Tab. 4.2, employing MobileNetV3 [59] as the backbone gives the best results across datasets. In the rest of the experiments of the following sections, we

Table 4.2: Comparison against state-of-the-art localization methods on 7Scenes dataset [132]. The first and second best results are marked in **bold** and underline, respectively. ‘_’ denotes unavailable results.

Method	Backbone	7Scenes							
		Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	Average
PoseNetGeo [69]	GoogLeNet	0.13/4.48	0.27/11.3	0.17/13.0	0.19/5.55	0.26/4.75	0.23/5.35	0.35/12.4	0.23/8.12
PoseNetLSTM [148]	GoogLeNet	0.24/5.77	0.34/11.9	0.21/13.7	0.30/8.08	0.33/7.00	0.37/8.83	0.40/13.7	0.31/9.85
GPoseNet [22]	GoogLeNet	0.20/7.11	0.38/12.3	0.21/13.8	0.28/8.83	0.37/6.94	0.35/8.15	0.37/12.5	0.31/9.95
BranchNet [159]	GoogLeNet	0.18/5.17	0.34/8.99	0.20/14.2	0.30/7.05	0.27/5.10	0.33/7.40	0.38/10.3	0.29/8.32
SVS-Pose [104]	VGGNet	-	-	-	-	-	-	-	-
Hourglass PN [95]	ResNet34	0.15/6.17	0.27/10.8	0.19/11.6	0.21/8.48	0.25/7.01	0.27/10.2	0.29/12.5	0.23/9.54
MapNet [20]	ResNet34	0.08/3.25	0.27/11.7	0.18/13.3	0.17/5.15	0.22/4.02	0.23/4.93	0.30/12.1	0.21/7.78
AtLoc [149]	ResNet34	0.10/4.07	0.25/11.4	0.16/11.8	0.17/5.34	0.21/4.37	0.23/5.42	0.26/10.5	0.20/7.56
CoordiNet [98]	ResNet34	-	-	-	-	-	-	-	-
CoordiNet [98]	EffNet b3	0.14/6.7	0.27/11.6	0.13/13.6	0.21/8.6	0.25/7.2	0.26/7.5	0.28/12.9	0.22/9.72
PoGO-Net [81]	Not Applicable	-1.72	-/6.23	-/7.34	-/3.93	-/3.56	-/3.85	-/7.88	-/4.93
MsTransformer [128]	EfficientNetB0	0.11/6.38	0.23/11.5	0.13/13.0	0.18/8.14	0.17/8.42	0.16/8.92	0.29/10.3	0.18/9.51
GNNPose [165]	ResNet34	0.08/2.82	0.26/8.94	0.17/11.41	0.18/5.08	0.15/2.77	0.25/4.48	0.23/8.78	0.19/6.32
Ours	ResNet34	0.05/1.86	0.14/4.15	0.09/5.29	<u>0.14/3.61</u>	0.11/2.78	0.10/2.66	<u>0.19/4.14</u>	0.12/3.50
	MobileNetV3	0.05/1.42	<u>0.16/4.57</u>	<u>0.09/6.14</u>	0.13/3.46	<u>0.12/2.48</u>	0.10/2.51	0.17/3.48	0.12/3.44

adopt MobileNetV3 [59] as our backbone while also providing a comparison with other backbones.

4.4.4 Results: Ablation Study

Here, we delve into our proposal by examining the effect of the three different losses that were employed and the performance of our method with different output resolutions and backbones. We list the results in Tab. 4.3.

Losses: we evaluate the contribution of every loss in our proposed method. Our method utilizes pose labels to guide the network through a pose loss Equation. (4.2). In addition, we apply consistency loss to align the two geometric representations (local and global) according to the input pose, and lastly, a re-projection loss to align 3D global coordinates to 2D image pixels. Tab. 4.3 (rows 1 to 4) presents the averaged results over all sequences on considered datasets, with different combinations of loss terms.

The results from Tab. 4.3 (rows 1 to 4) show that adding the re-projection loss (row 2) or the consistency loss (row 3) to the pose loss (row 1) results in lower errors than when training the network with the pose loss only (row 1, rigid alignment module). The consistency loss is more effective than the re-projection loss in supporting the pose loss to reduce localization errors on outdoor scenes. On indoor scenes, they exhibit almost similar behavior. Combining all the losses (row 4) obtains the lowest errors on all datasets.

Resolutions: We evaluate the performance of our proposed method with different

output resolutions. In our experiments, the resolution of the output geometric representations is 1/8 of the input resolution. Here, we change the output resolution by changing the stride parameter. We report the results of two additional down-sampling factors: 4 and 16 in Tab. 4.3 (rows 5 to 7). For a down-sampling factor of 16, we observe a slight increase in localization errors compared to a down-sampling factor of 8. On average, down-sampling by a factor of 4 results in a slight improvement in localization.

Depth versus 3D Camera coordinates: We can adjust the network to either obtain 3D coordinates in the camera frame directly or obtain the depth. For the latter, 3D camera coordinates can be computed by Equation. (4.6). Rows 8 and 9 in Tab. 4.3 suggest that learning just the depth results in a better localization performance. This constrains the 3D coordinates predictions and eases the learning process so that the network learns one quantity rather than three quantities.

Backbones: While many backbones could be used to implement our method, we look into them from the perspective of localization accuracy, run-time, and compactness. The rows 10 to 13 in Tab. 4.3 show the results using different backbones together with the run-time. Being the most compact, that is, with the smallest number of parameters (3.7 million), MobileNetV3 [59] obtains the best localization results with the fastest run-time (for a down-sampling factor of 8).

Table 4.3: Ablation results of section 4.4.4 on Cambridge Landmarks [70], 7Scenes [132], and 12Scenes [147] datasets.

			Cambridge	7Scenes	12Scenes
Losses					
	L_{pose}	$L_{reprojection}$	$L_{consistency}$		
1	✓			0.72/3.91	0.138/3.88
2	✓	✓		0.71/2.89	0.119/3.49
3	✓		✓	0.67/1.75	0.118/3.49
4	✓	✓	✓	0.60/1.62	0.116/3.44
Resolutions					
5	Input Resolution/4		0.59/1.62	0.115/3.13	0.060/2.31
6	Input Resolution/8		0.60/1.62	0.116/3.44	0.061/2.33
7	Input Resolution/16		0.68/1.77	0.134/4.15	0.068/2.65
Depth versus 3D Camera coordinates					
8	Ours + 3D Coordinates		0.64/1.85	0.131/4.03	0.064/2.57
9	Ours + Depth		0.60/1.62	0.116/3.44	0.061/2.33
Backbones					
	Backbone	Run-time			
10	HRNetV2 [151]	256 ms	0.64/2.01	0.131/3.78	0.068/2.61
11	ResNet34 [57]	45 ms	0.63/1.74	0.124/3.50	0.064/2.49
12	DenseNet121 [61]	90 ms	0.62/1.69	0.121/3.52	0.062/2.43
13	MobileNetV3 [59]	14 ms	0.60/1.62	0.116/3.44	0.061/2.33

4.4.5 Results: Outliers Filtering

Our method estimates weighting factors for each 3D-3D correspondence. These account for imperfections in 3D coordinates predictions and aim to down-weight 3D correspondences that lead to inferior localization results (i.e., outliers). We conduct experiments to assess the impact of these weighting factors. Since our method relies on a single image for localization, we consider the following methods:

Rigid + No Filtering: We compute pose using rigid alignment without incorporating the obtained weights. That is, all predicted 3D correspondences are of equal importance.

Rigid + Filter Dynamic: We utilize semantics to filter out contributions from sources of outliers, mainly dynamic objects. Specifically, we use the recently released Intern-Image [153] to segment the input image into semantic classes. We filter out 3D points that correspond to pixels of dynamic classes. These are pedestrians, cars, bicycles, and trucks.

Rigid + Filter Dynamic + Others: We complement the removal of dynamic points by pruning 3D points that correspond to semantic classes that are presumably inferior to localization, such as sky and trees.

Rigid + RANSAC: We utilize a robust outlier filter by pairing the rigid alignment algorithm with a RANSAC scheme [43]. We apply RANSAC with a maximum of 2000 iterations and an inlier threshold of 10 cm between corresponding 3D points. We use 10 correspondences for pose estimation.

PnP + RANSAC: Our method uses poses to obtain 3D coordinates in the global coordinate system of the scene. Besides utilizing rigid-alignment, our method offers the flexibility to adopt perspective-n-point (PnP) algorithm for pose estimation using the 3D global coordinates and the corresponding 2D pixels. We compute pose using PnP [48] from 4 correspondences using RANSAC [43]. We set 2000 as the maximum number of RANSAC iterations, with an inlier threshold of 10 pixels.

Rigid + Weights [ours]: We compute pose using weighted rigid alignment where the weights are the ones obtained by our method. These are obtained directly with a forward pass of the network without further processing of the obtained 3D correspondences or any off-the-shelf outlier filter.

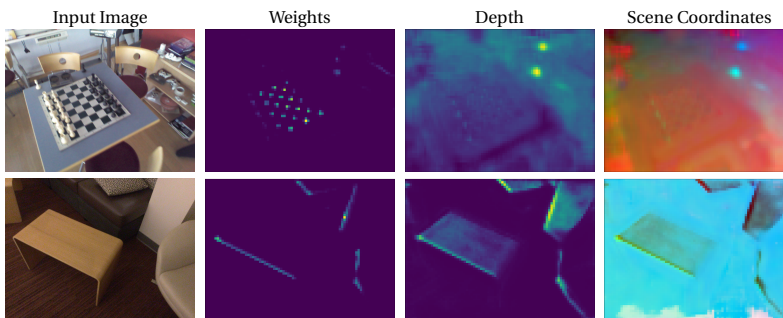
Tab. 4.4 lists the results on Cambridge Landmarks [70]. Pruning off 3D points that correspond to dynamic objects by semantic segmentation shows improvements in localization compared to utilizing all the 3D points. In addition to removing dynamic points, filtering out correspondences from non-informative regions like sky and trees obtains additional improvements. The hospital scene does not contain dynamic objects which justifies the same result that is obtained without filtering any 3D-3D correspondence. While semantics can be used to filter complete dynamic objects, it might not filter all sources of outliers. Besides, it is subject to segmentation errors and may filter many useful features. Pairing a robust outlier filter such as RANSAC [43] with both

Table 4.4: Effect of different filtering methods (section 4.4.5). The best results are marked in bold.

Method	College	Hospital	Shop	Church
Rigid + No Filtering	0.55/0.87	0.79/1.63	0.59/2.32	1.13/3.55
Rigid + Filter Dynamic	0.53/0.86	0.79/1.62	0.56/2.27	1.06/3.49
Rigid + Filter Dynamic + Others	0.52/0.93	0.76/1.51	0.50/2.19	1.01/3.46
PnP + RANSAC	0.49/1.04	0.58/1.64	0.41/2.72	1.45/4.72
Rigid + RNASAC	0.46/1.13	0.60/1.73	0.46/3.30	1.12/3.58
Rigid + Weights [ours]	0.46/0.81	0.61/ 1.09	0.44/ 1.71	0.87/2.88

PnP [48] and rigid alignment shows further improvements in estimated translation. However, it leads to increased errors on the Church scene and elevated orientation errors overall. We trace this back to the reason that many 3D points may be inferior and that few of them are subject to inlier checks. The Church scene poses a difficult localization scenario where the camera moves 360° around the scene. To overcome this and improve further, it would require increasing the number of considered points, relaxing the inlier threshold, and/or increasing the maximum number of iterations. In summary, utilizing RANSAC [43] for filtering outliers requires adjusting hyperparameters for each scene besides imposing a run-time burden.

In contrast to the mentioned methods, using the set of weights that are obtained by our method delivers a consistent reduction in position and orientation localization errors. These weights form a prior about the useful 3D coordinates for localization. It considers all correspondences for localization, nevertheless, with different weights. We complement these quantitative observations by showing visualizations of the network outputs in Fig. 4.3.

**Figure 4.3:** Visual samples of the predictions obtained by the network on samples from 7Scenes. For visualizing the 3D coordinates, we map the X , Y , Z coordinates to RGB values.

Inspecting the visualization, we observe that the network guesses the scene's architecture from pose labels only. Furthermore, the weights' heatmaps show that the

network focuses on a small subset of features, mainly edges, and corners, implying that the correct geometric predictions lie on these features. It estimates the depth and 3D scene coordinates without supervised labels for these quantities.

4.4.6 Results: Finetuning With Position Labels Only

In this section, we test our method with limited training samples and explore the possibility of using only partial labels (only position information and not orientation). These partial position-only labels can be easily available from GPS or can be available on the fly using another sensor. Testing with limited data is challenging for deep learning-based methods as they require more data for generalization. Accordingly, we sample a third of the dataset for training by taking every third training sample. The testing samples are kept in the original size as provided by the dataset. After training, we finetune the network for a few epochs (10-15) by feeding it with the other two-thirds of the training samples. However, we assume that these additional training samples have only partial geometric labels (translation only instead of 6 DoF poses). Thus, we apply only the translation loss for finetuning from Equation. (4.3). The consistency (Equation. (4.5)) and re-projection (Equation. (4.7)) losses are excluded as they require full 6 DoF poses. In addition, we apply the same training scheme (training with absolute position labels) on one of the state-of-the-art pose regression methods: MapNet [20].

Table 4.5: Results of the experiment of section 4.4.6 on Cambridge Landmarks [70]. Median errors (meters/degrees) are reported. Improvements as a result of finetuning by $L_{position}$ over training on 1/3 of samples are marked by underlines.

Method	Training Scheme	Cambridge			
		College	Hospital	Shop	Church
Ours	trained on all samples	0.46/0.81	0.61/1.09	0.44/1.71	0.87/2.88
	trained on 1/3 of samples	0.52/0.85	0.69/1.49	0.59/2.53	0.95/3.04
	finetuned with $L_{position}$	<u>0.50/0.88</u>	<u>0.68/1.32</u>	<u>0.59/2.44</u>	<u>0.93/2.95</u>
MapNet [20]	trained on all samples	0.94/1.99	2.03/3.60	0.80/6.34	1.66/4.01
	trained on 1/3 of samples	1.12/6.10	3.14/7.81	1.29/8.76	2.65/6.54
	finetuned with $L_{position}$	1.25/93.02	3.03/139.79	1.34/95.01	2.63/53.50

The results are listed in Table 4.6. As expected, the localization accuracy dropped when training with fewer samples. MapNet [20] yields a larger drop in accuracy than our method. Our method reports a little reduction in accuracy on indoor scenes. Finetuning the model given only position labels has reduced both the translation and rotation errors on some of the scenes. In contrast, MapNet [20] has a significant increase in orientation errors when finetuned with position labels. The reason behind the better performance of our algorithm is driven by two correlated reasons. The first is the separation between image representations and pose estimation that is inherent to our algorithm, while the second being the computation of the pose using non-learned and parameter-free rigid-alignment, which updates both 3D point clouds from position supervision. The rigid alignment module passes gradients to all the network branches

Table 4.6: Results of the experiment of section 4.4.6 on 7Scenes dataset [132]. Median errors (meters/degrees) are reported. Improvements as a result of finetuning by $L_{position}$ over training on 1/3 of samples are marked by underlines.

Method	Training Scheme	7Scenes						
		Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
Ours	trained on all samples	0.05/1.42	0.16/4.57	0.09/6.14	0.13/3.46	0.12/2.48	0.10/2.51	0.17/3.48
	trained on 1/3 of samples	0.06/1.60	0.18/4.73	0.13/7.7	0.14/3.48	0.11/2.73	0.12/2.81	0.19/3.50
	finetuned with $L_{position}$	<u>0.05/1.55</u>	0.18/5.28	0.13/7.7	<u>0.12/3.41</u>	0.12/3.13	<u>0.11/2.63</u>	<u>0.18/3.74</u>
MapNet [20]	trained on all samples	0.08/3.25	0.27/11.7	0.18/13.3	0.17/5.15	0.22/4.02	0.23/4.93	0.30/12.1
	trained on 1/3 of samples	0.12/4.75	0.30/11.51	0.18/14.02	0.20/6.20	0.21/5.33	0.27/5.76	0.37/11.53
	finetuned with $L_{position}$	0.12/38.98	0.31/24.68	0.18/20.28	0.19/41.52	0.22/35.62	0.26/32.00	0.37/24.23

that predict the geometric quantities. Finetuning using only positional labels works well. While the accuracy did not reach that of training on the complete dataset, the improvements for both location and orientation given only translation labels open doors for further research in this direction.

4.4.7 Results: Run-time

In Table 4.7, we report the run-time of our method for different down-sampling factors (resolutions) on an input with a standard resolution 480×640 . We run our Python implementation on a machine equipped with GTX Titan X and Intel Core i7-5960X CPU @ 3.00GHz. The results imply that our method localizes in real run-time. We also report the run-time of a minimal pose regression pipeline (PoseNet [70]) using MobileNetV3 backbone [59]. Some state-of-the-art regression methods further process the output of the network before pose regression by applying attention [149], graph neural networks [165], and transformers [128], demanding additional run-time.

Table 4.7: Run-time analysis (section 4.4.7). Data Processing: time needed to run the network and obtain the 3D clouds. Pose Computation: time needed to obtain pose through rigid alignment. FPS: frames per second.

	Down-sampling Factor	Output Resolution	Data Processing	Pose Computation	Total (ms)
					FPS
Ours	4	120×160	9.0 ms	30.0 ms	39.0 ms - 26 Hz
	8	60×80	9.2 ms	4.8 ms	14.0 ms - 71 Hz
	16	30×40	9.6 ms	1.5 ms	11.1 ms - 90 Hz
Regression					12.5 ms - 80 Hz

4.5 Conclusions

We presented a novel global 6 DoF pose estimation method from a single RGB image. The proposed work shares with most existing pose regression methods the same constraints, which are: train from a set of image-pose pairs, estimate a pose from a single image, save only the weights of the network, and obtain a pose in real run-time. However, our method obtains more accurate pose estimates, as we have shown on common public datasets. The reason stems from the incorporation of scene geometry into pose estimation. The difficulty in achieving that, nonetheless, lies in the utilization of the only given labels (poses) to estimate this geometry and the use of the geometry for real run-time pose estimation. Our method's main novelty is using pose targets only to guide a deep neural network through differentiable rigid alignment to estimate the scene geometry without explicit ground-truth of this geometry at training time. The proposed method takes a single image and implicitly obtains geometric representations of the scene using only pose labels. These implicitly learned geometric representations are the 3D scene geometry (X , Y , Z coordinates) in two reference frames: global and camera coordinate systems. We utilize a parameter-free and differentiable rigid alignment to pass gradients through a deep neural network to adjust its weights and continually learn these representations without explicit ground-truth labels. Besides pose loss, another novelty is that our method allows for the inclusion of additional learning losses as opposed to learning a localization pipeline by pose regression. We introduce a consistency loss to make the two geometric representations consistent with the geometric pose and a re-projection loss to constrain the 3D global coordinates to the 2D image pixels. Through extensive experiments, we show that the proposed method exceeds the localization accuracy of state-of-the-art regression methods and runs in real-time. As a final contribution, we show that the proposed formulation can utilize partial labels (instantaneous position labels only) to finetune a pre-trained model, leading to improvements in both position and orientation localization. In future, we would like to leverage foundational models such as SAM [72] and CLIP [107] to generate embeddings and integrate them into our learned 3D representations to perform more accurate pose estimation using scene semantics.

Chapter 5

Long-Term Invariant Local Features via Implicit Cross-Domain Correspondences[†]

Modern learning-based visual feature extraction networks perform well in localizing images within the same environment. However, their performance drops significantly when attempting to match images captured under different long-term visual conditions, such as seasonal and daytime variations. In this paper, our first contribution is a benchmark aimed at assessing the impact of long-term variations on visual localization performance. We thoroughly analyze the effectiveness of current state-of-the-art feature extraction networks under various domain variations and observe a substantial performance gap between matching images in the same domain and across different domains. We explore various methods to bridge this performance gap by enhancing the supervision of modern feature extraction networks. We introduce a novel data-centric method called Implicit Cross-Domain Correspondences (iCDC). iCDC employs multiple Neural Radiance Fields to represent the same environment under various visual domains, using the underlying 3D representations to accurately match images across different long-term visual conditions. Our proposed method enhances cross-domain localization performance, significantly narrowing the performance gap. When evaluated on popular long-term localization benchmarks, our trained networks consistently outperform existing methods. This research represents a significant step towards more reliable visual localization systems for long-term deployments and lays the groundwork

[†]This chapter is based on the work "Long-Term Invariant Local Features via Implicit Cross-Domain Correspondences". The work is under review for IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) and available on arXiv preprint, arXiv:2311.03345, Nov. 2023. The article has been modified to fit the format and requirements of this dissertation.

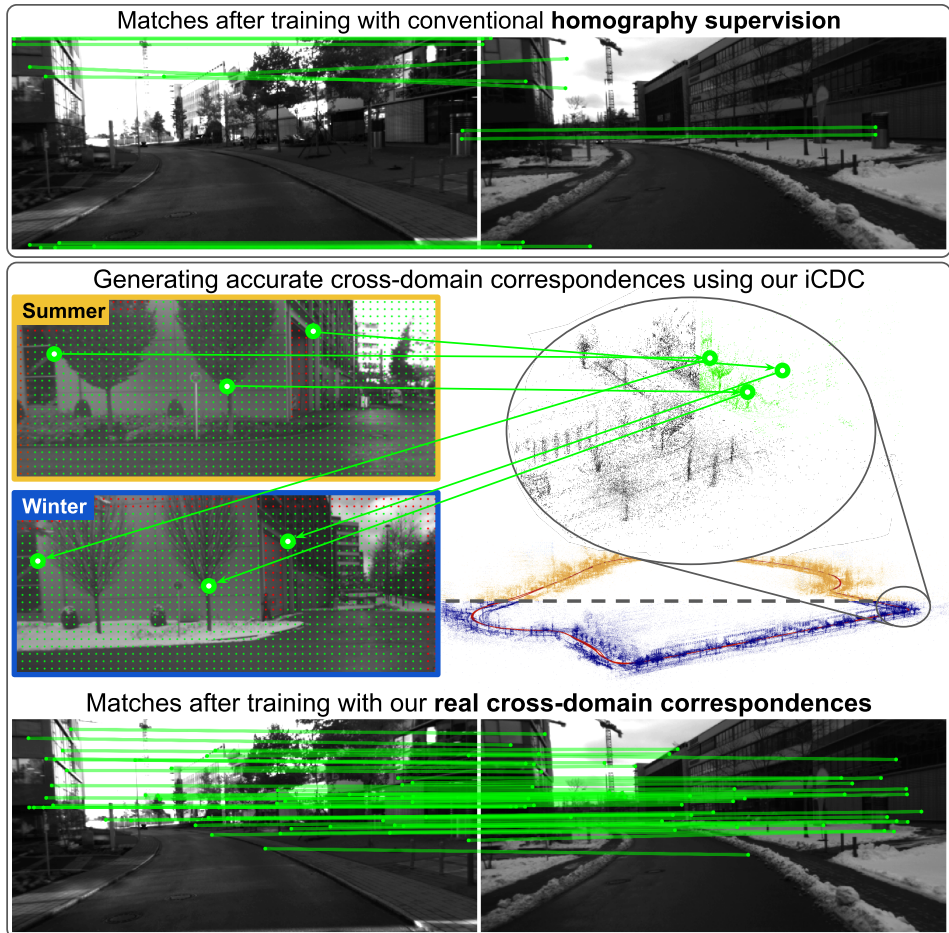


Figure 5.1: We propose iCDC, a method to supervise feature extractors across different domains. Traditionally, correspondences are generated by augmenting and applying homographic transformations to images. iCDC generates accurate correspondences across images captured under different views and visual domains.

for developing long-term invariant descriptors.

5.1 Introduction

Visual localization is essential for many applications, such as guiding autonomous vehicles and integrating virtual objects into the real world. Data association plays a criti-

cal role in solving the visual localization problem. It involves detecting and describing significant points in images to establish correspondences between keypoints. These correspondences help estimate the relative pose of images from different viewpoints, enabling 3D scene reconstruction and determining the camera pose of a query image relative to a 3D map.

The quality of data association heavily relies on the accuracy and robustness of local feature extraction and description. Initially, sparse features were identified using hand-crafted methods [88, 11, 78, 116], which have seen success in various applications, including SfM [126] and Simultaneous Localization and Mapping (SLAM) [101]. However, significant advancements in Deep Neural Networks (DNNs) have enabled the learning of robust and highly descriptive image features [34, 114, 39, 66, 50], pushing the boundaries of what was previously possible [124, 63, 9].

In order to achieve successful deployment in real-world scenarios over the long-term, it is important to evaluate the robustness of visual systems to changes caused by time of day, weather, and season. Benchmarks such as long-term localization [124] or sparse SLAM under different weather conditions [156] can be used for this evaluation. In life-long deployment scenarios, accurate six Degrees of Freedom (DoF) localization depends on localizing with respect to a previously built 3D map. The main challenge here arises from the fact that the visual characteristics of the descriptors in the map and those in the query often differ. While model-centric methods are important, we believe that further exploration of data-centric methods is necessary to achieve more accurate localization performance across long-term variations.

In this work, we aim to understand and improve the robustness of feature extraction networks in handling long-term visual changes. To thoroughly assess long-term localization performance, we have improved the 4Seasons dataset [156] by refining the poses. This dataset contains a significant amount of data captured across various visual domains and scenes, but it lacks consistency in camera poses across different trajectories. By refining poses through cross-trajectory and cross-domain optimization, we can accurately compare the relative poses of frames across different scenes. With our refined maps, we evaluate the performance of current extractors [34, 114, 39, 139, 50].

Our experiments reveal a significant performance gap between intra-domain and cross-domain localization, regardless of the domain types. To bridge this gap, we explore and categorize data-centric approaches by supervising feature extractors using cross-domain correspondences. Through a comparative analysis, we identify the limitations of the approaches and propose a novel data-centric method, Implicit Cross-Domain Correspondences (iCDC).

iCDC represents an environment using multiple, independent, and domain-specific Neural Radiance Fields (NeRFs) and leverages their underlying 3D representations to generate highly accurate cross-domain correspondences. We train state-of-the-art networks [114, 50] using iCDC and improve their cross-domain localization performances on our benchmark and on other long-term localization benchmarks [125, 90, 8].

In summary, our contributions are the following:

- We refine the ground-truth maps of 4Seasons and establish a new benchmark for long-term localization.
- Our benchmarking of existing feature extraction networks reveals a significant performance gap between intra- and cross-domain localization.
- We find that supervising extractors using cross-domain correspondences between real viewpoint changes reduces the performance gap.
- We propose iCDC to generate accurate cross-domain correspondences. Training extractors using iCDC further improves cross-domain localization.

5.2 Related Work

Hand-crafted feature extractors. Hand-crafted extractors such as SIFT [88], SURF [11], BRISK [78], BRIEF [23], ORB [116] aim to identify and describe distinctive structures or patterns in images. SIFT is well-regarded for its invariance to scale, rotation, and affine transformations and has been widely used in applications such as object recognition [88] and SfM [126]. Inspired by SIFT, SURF improved the computation speed of the feature extraction while maintaining robustness to scale and rotation changes by approximating the Laplacian of Gaussian with a box filter representation. On the other hand, some methods [78, 23, 116] focus instead on the generation of binary descriptors for use in computationally limited platforms such as robots and Augmented Reality (AR). While these hand-crafted methods have been successful in a range of applications, they often struggle with perceptual changes caused by variations in illumination, weather, and seasons [124].

Learned feature extractors. Motivated by the limitations of hand-crafted methods and the success of DNNs, learning-based feature extractors [34, 114, 39, 66] have improved performance, more noticeably in the aforementioned challenging perceptual changes [124]. A number of works explored better architectural designs to enhance performance. Notably, R2D2 [114] and D2-Net [39] propose networks that share all parameters between detection and description. SuperPoint [34], and KP2D [139], jointly learn interest point detection and description using two decoding branches. Furthermore, other works aimed to improve the optimization of the networks through outlier rejection [139], introducing a novel reliability score to guide the detection process [114], contrastive learning [28], using guidance from SfM to obtain more robust keypoints [3], or by employing reinforcement learning principles [144]. Recent works [66, 42] investigated the possibility of making these networks more efficient for deployment in computationally limited platforms, while others [50] focused on improving retrainability; a limitation of DISK [144] which is notoriously difficult to train.

A common component amongst these works, with the exception of DISK, is the use of self-supervised methods, which facilitate the flexible optimization of the networks, enabling learning from diverse and large-scale datasets, such as COCO [84]. This is accomplished by first applying non-geometric transformations, such as color augmentations, as well as a homographic transformation to every input image, and then optimizing the network to generate identical descriptors for the same regions in the original and transformed images. While this method has proven very powerful in practice, invariance to perceptual changes caused by weather and seasonal variations is not directly enforced, potentially hindering their performance under challenging perceptual changes caused by the time of day, season, and weather conditions. In this work, we aim to evaluate the robustness of state-of-the-art extractors in both intra- and cross-domain scenarios, exploring avenues for further improvement.

Invariance to long-term changes. Due to the importance of the lifelong deployment of computer vision systems, a number of different approaches have been proposed over the years to close the cross-domain performance gap for different applications. This includes optimizing networks by directly supervising cross-domain information through contrastive learning [6, 26, 27]. This has been explored in image retrieval and place recognition networks that output a single descriptor to represent the image, where generating ground-truth labels is made possible through GPS signals available in large datasets [157, 6]. For a number of applications, however, generating ground-truth labels is expensive. This has given rise to domain adaptation [41, 32, 31] and generalization [80, 172] methods, allowing the optimization of networks on datasets where labeled data can be more easily acquired, such as simulations, while minimizing the domain gap with real-life data. This technique has been widely adopted in semantic segmentation [131, 55, 129, 160, 21] and image classification [145, 67, 45] tasks.

More similar to our work, WarpC [141] employs an unsupervised warping consistency to improve correspondence learning. In contrast, other methods [5, 114] use style transfer [94] instead, to alter the domain of the images, such as converting a summer image into a winter image. We, on the other hand, propose to directly enforce cross-domain similarity of descriptors by leveraging a large-scale dataset that captured the same trajectories at different weather conditions [156]. This is facilitated through our novel method iCDC, a correspondence generation method for real cross-domain images.

Long-term datasets. Existing research has shown feature extractor methods to be highly effective for accurate localization within the same domain [124, 117]. A growing body of literature seeks to address the challenge of long-term localization in situations that require matching descriptors across different domains. Benchmark datasets like the Aachen Day-Night dataset [125] focus on localizing query night-time images with respect to a day-time map, while RobotCar Seasons [90] benchmarks visual localization across an array of query domains against a single reference map capturing a single domain. Notably, its images are of relatively low quality, with night-time images suffering from significant motion blur. On the other hand, CMU-Seasons [8] benchmarks localization across many visual domains, albeit primarily in rural areas, with large portions

of the images capturing vegetation. Although these benchmarks contribute to the research on cross-domain localization, the range of domain pairs and scenes examined remains limited.

On the other hand, the 4Seasons dataset [156] encompasses data across multiple trajectories, captured under various weather and seasonal conditions and in diverse scenes across different environments. However, it lacks reliable relative poses between images captured along different trajectories. We jointly optimize the trajectories to provide accurate relative poses across a broad spectrum of visual domains to establish it as a new long-term localization benchmark.

5.3 Optimizing the 4Seasons Dataset to Benchmark Cross-Domain Visual Localization

5.3.1 Original 4Seasons Dataset Details

Table 5.1: Dataset overview: trajectory labels are used in this work to refer to the corresponding trajectories. The domain is described for each trajectory. For each scene, the average number of frames with RTK-GNSS measurements per trajectory is provided, as well as the average of the total number of available frames.

Trajectory Labels	Season	Weather Condition	Time of Day
Business Campus (3255 RTK-GNSS poses, 11765 available frames)			
BC1	Fall	Sunny	Morning
BC2	Winter	Cloudy/snowy	Afternoon
BC3	Winter	Sunny	Afternoon
Office Loop (2775 RTK-GNSS Poses, 14438 available frames)			
OL1	Spring	Sunny	Afternoon
OL2	Spring	Sunny	Afternoon
OL3	Spring	Sunny	Morning
OL4	Summer	Sunny	Morning
OL5	Winter	Cloudy/snowy	Afternoon
OL6	Winter	Sunny	Afternoon
Neighborhood (2790 RTK-GNSS poses, 10297 available frames)			
N1	Spring	Cloudy	Afternoon
N2	Fall	Cloudy	Afternoon
N3	Fall	Rainy	Afternoon
N4	Winter	Cloudy	Morning
N5	Winter	Sunny	Afternoon
N6	Spring	Cloudy	Evening
N7	Spring	Cloudy	Evening

Sattler et al. [124] established the benchmark for long-term visual localization by building on existing datasets such as Aachen Day-Night [125], Oxford RobotCar [90], and CMU Seasons [8]. The autonomous driving 4Seasons dataset [156] provides more

diverse data with accurate ground-truth poses across various weather, seasonal, and day-time conditions in a larger number of environments compared to the existing datasets in this benchmark.

To create a cross-domain localization benchmark, we required a dataset with a wide range of visual condition variations, which the 4Seasons dataset met. Additionally, for a comprehensive analysis, the dataset needed accurate ground-truth relative poses across individual trajectories (visual domains). A dataset meeting these criteria would allow the evaluation of localization performance across any two visual conditions within a scene.

In the 4Seasons dataset, Real-time Kinematic and Global Navigation Satellite System (RTK-GNSS) measurements were utilized for a subset of frames to enhance the accuracy of the ground-truth poses (More information can be found in the 4Seasons paper [156]). These measurements also provide geolocations for each frame, allowing individual trajectories to be represented within a unified coordinate system. However, the dataset’s pose optimization was only done along individual trajectories, leading to unreliable cross-trajectory poses and inaccurate maps. As a result, the current maps in the 4Seasons dataset do not meet our requirements. The unreliable relative poses cause misaligned 3D representations of the maps, making them unsuitable for our data-centric supervision approach. This highlights the need for better maps.

5.3.2 Refining the 4Seasons Dataset

We adhere to the HLoc [117] pipeline, which integrates and improves upon COLMAP [126], to construct SfM maps meeting our criteria for selected scenes from the 4Seasons dataset. Table 5.1 provides an overview of the data from these scenes, including trajectory labels referenced later in this paper. We utilize the available poses in the 4Seasons dataset to expedite the map optimization process as an initialization step. A summary of our refinement pipeline is depicted in Figure 5.2.

Map initialization. We initialize our maps by using the provided poses of all frames from a 4Seasons scene across multiple trajectories within a unified coordinate system. Frames with registered geolocations, i.e., with RTK-GNSS positions, are placed in this unified coordinate system and act as anchors for map initialization. For the other frames, poses are available only in a local coordinate system, optimized through stereo visual-inertial odometry. Despite suffering from drift, we utilize these locally accurate relative poses to initialize the global map by positioning them around the anchor frames.

Image matching. In the first step of the HLoc pipeline, covisible database images are identified. We generate these covisible image pairs using the ground-truth relative poses available in our initialized maps. For each frame, we locate the nearest n frames both within and across trajectories.

SfM map triangulation. In the next stage, COLMAP triangulates shared keypoints

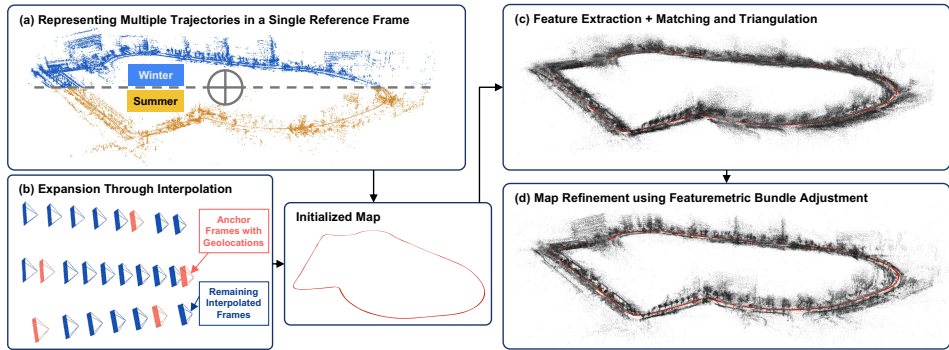


Figure 5.2: Joint trajectory map refinement. (a) Frames with registered geolocations along multiple trajectories of the 4Seasons dataset are placed as anchors into a global map. (b) The remaining frames have poses only in a local reference frame optimized through stereo visual-inertial odometry. They are integrated into the map by interpolating them around the anchor frames. (c) SuperPoint [34] features are extracted and matched using SuperGlue [118]. Using the poses in the initialized map, the matched keypoints are triangulated into 3D. (d) Pixel-Perfect-SfM [85] is leveraged to optimize the 3D points and poses in the map through feature-metric bundle adjustment.

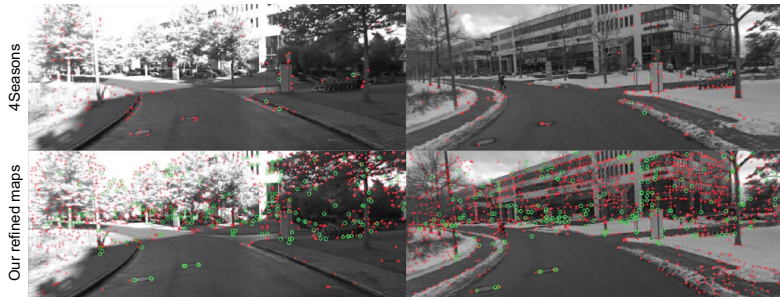


Figure 5.3: The figure showcases keypoints and matches from the 4Seasons dataset and from our refined maps. Matches across frames are highlighted in green. For the 4Seasons dataset, keypoints are matched using corresponding depth values and frame poses, while matches from our refined maps are directly extracted from the maps.

across images into 3D. While the provided 4Seasons poses can initialize the maps, the provided keypoints are not reliable for global map optimization. This is due to the limited number of keypoints available and the absence of keypoint matches across image pairs in the dataset. Although combining per-keypoint depth values with relative poses could theoretically enable geometric keypoint matching, the unreliability of relative poses across trajectories renders these geometric matches unreliable. Instead, we detect keypoints using SuperPoint [34], match them across identified image pairs using SuperGlue [118], and use them for triangulation. Figure 5.3 compares the sparse set of cross-domain 4Seasons keypoints and geometric matches with the abundance of SuperPoint + SuperGlue matches in our maps. SuperPoint + SuperGlue generates

accurate matches across domains [125], leading to highly accurate maps when used in a Structure-from-Motion (SfM) pipeline.

Global map optimization. In the final stage of the HLoc pipeline, maps undergo global optimization through bundle adjustment. Traditionally, COLMAP is used to refine poses and 3D points in the map by globally minimizing reprojection errors. However, leveraging the latest advances in SfM, we employ Pixel-Perfect-SfM (PixSfM) [85]. Unlike COLMAP, the global refinement in PixSfM is termed "featuremetric" because it optimizes the consistency of deep features across frames and refines keypoints. Instead of relying on reprojection errors, PixSfM optimizes 3D points, keypoints, and poses by minimizing a featuremetric cost.

5.3.3 Refined Map Quality Analysis

As there are no definitive "true" ground-truth poses, we are unable to measure the actual accuracy of our maps. However, we demonstrate the improved quality of our maps over the ground-truth poses available in the 4Seasons dataset. We present both qualitative and quantitative analyses. While we calculate relative poses for all frame pairs in a scene, for this study, we limit our analysis to the frames with registered geolocation positions. The other frames in the 4Seasons dataset exist in their own local coordinate system.

Qualitative map quality analysis. We project keypoints from a query image onto a reference image using the estimated depths and relative poses from the corresponding 3D map. The quality of the maps can be visually evaluated by studying the reprojection of keypoints of a static object, such as a building. Errors in the map become apparent when the reprojected points on the query image do not match the same points in the reference image. These discrepancies can stem from inaccuracies in both depth and pose. However, unlike depth errors, pose errors result in consistent reprojection errors across all keypoints in the image pair.

In Fig. 5.4, we showcase cross-domain pairs of reference and query images that demonstrate the reprojection of keypoints of static objects and reveal relative pose errors in the 4Seasons dataset. These are noticeable and can be clearly seen along the edges of the static objects (we highlight some with cyan lines). In contrast to reprojecting using keypoints and poses from the original 4Seasons dataset, reprojection using the refined poses yields more accurate correspondences.

Quantitative map quality analysis. In the next section, we additionally benchmark the cross-domain localization performances of state-of-the-art feature extraction networks using our refined poses as ground-truth and properly introduce the metrics. We summarize the results in Tab. 5.2, comparing these with localization results evaluated against the 4Seasons poses treated as ground truth, presented in Tab. 5.3. We report the median localization errors and, following [118, 135], Area Under the Curve (AUC) of pose errors at error thresholds: 5° , 10° , and 20° , averaged across all scenes and tra-

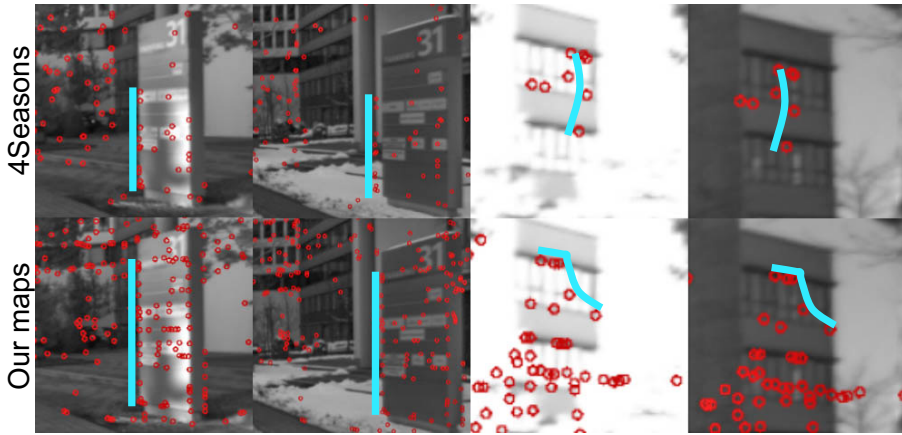


Figure 5.4: Keypoint reprojection for pose quality analysis. Each row presents image pairs captured during different seasons. Keypoints from each left-side image are reprojected onto the right-side image using ground-truth poses and keypoint depths. The top and bottom rows are reprojected keypoints from the 4Seasons maps and our maps, respectively. Lines have been drawn to highlight references and corresponding reprojected keypoints. The projected keypoints along the highlighted lines reveal consistent reprojection errors across the 4Seasons frame pairs. These consistent errors indicate inaccurate relative poses. The same can not be observed in the case of our refined maps.

jectory pairs.

All feature extractors achieve higher performances, i.e., lower median errors and higher AUC under all thresholds, when benchmarked against our refined poses. Although extractor performance is not a perfect indicator of improved map accuracy due to inherent imperfections, the consistent reduction in localization errors underscores the improvements achieved through our refinement process. While our primary goal was to enhance cross-domain poses, the experiment also demonstrates improvements in ground-truth poses within individual trajectories.

The combined qualitative and quantitative analyses indicate that the refined poses can effectively evaluate cross-domain localization accuracy. This new dataset will be used for the remainder of the paper. With the capability to assess localization performances across all visual domains available in each scene, our presented benchmark offers an opportunity for the most comprehensive study of cross-domain localization to date.

Table 5.2: Relative pose errors evaluated on 4Seasons poses vs. our refined poses across different illumination conditions. Pose errors are presented both as median errors and AUC accuracies. Consistent higher performances when evaluated against our refined poses indicates higher ground-truth pose quality.

		Cross Domain				
		SuperPoint [34]	R2D2 [114]	D2-Net [39]	KP2D [139]	SILK [50]
Median Error (°) ↓	4seasons	9.94	13.07	10.13	12.44	17.98
	Refined	9.78	12.65	9.89	12.23	17.84
AUC (%) ↑	4seasons	27.4/41.8/54.5	28.4/41.8/52.9	21.8/37.4/52.1	26.5/40.1/51.9	23.1/35.6/47.2
	Refined	29.1/43.0/55.2	30.6/43.5/54.1	22.8/38.1/52.5	28.2/41.3/52.6	24.5/36.4/47.6

Table 5.3: Relative pose errors evaluated on 4Seasons poses vs. our refined poses under the same illumination conditions. Pose errors are presented both as median errors and AUC accuracies. Consistent higher performances when evaluated against our refined poses indicates higher ground-truth pose quality.

		Intra Domain				
		SuperPoint [34]	R2D2 [114]	D2-Net [39]	KP2D [139]	SILK [50]
Median Error (°) ↓	4seasons	1.05	0.90	1.57	1.00	0.97
	Refined	0.96	0.81	1.49	0.91	0.88
AUC (%) ↑	4seasons	71.6/84.4/91.5	76.2/87.4/93.4	60.7/77.7/87.5	73.9/85.7/91.9	74.5/86.1/92.3
	Refined	73.7/85.7/92.3	78.5/88.7/94.1	62.4/78.7/88.1	76.0/86.9/92.7	76.3/87.0/92.7

5.4 Benchmarking Localization Robustness Across Visual Domains

Utilizing our refined maps, we compare state-of-the-art feature extraction methods for long-term localization. We assess localization performance using both relative and absolute pose estimation.

5.4.1 Benchmarking Relative Pose Estimation

To benchmark feature extractors, we propose matching keypoints between image pairs, determining the relative pose between the two frames, and assessing the accuracy of the predicted pose. The estimated translation is determined only up to a scale factor and is, therefore, reported as a normalized direction vector. Rotation and translation pose errors are measured as the relative angles between the estimated and ground truth rotation matrices and translation vectors. In line with previous works [118, 135], we report the pose errors, in degrees, as the maximum of translation and rotation angular errors.

Keyframing. We select a set of reference frames and corresponding query frames for each scene and each trajectory pair. Reference frames are selected at 10-meter intervals along the reference trajectory. For each chosen reference frame, we identify corre-

sponding query frames from the set of frames within 8 meters and 45° of the reference frame. These query frames are selected at 2-meter intervals.

Table 5.4: Median relative pose errors on the 4Seasons dataset [156]. For each scene, aggregated median errors are presented and evaluated against our ground truth poses. In the case of cross-domain, the error under reference trajectory is the mean of all median pose errors between the reference and all query trajectories.

Scenes	Business Campus			Office Loop						Neighborhood							ALL	
	Traj.	BC1	BC2	BC3	OL1	OL2	OL3	OL4	OL5	OL6	N1	N2	N3	N4	N5	N6		N7
Intra-domain																		
SuperPoint [34]	1.28	1.07	1.26	1.06	1.02	0.95	0.91	0.90	0.94	0.85	0.78	0.89	0.90	0.79	0.85	0.88	0.96	
R2D2 [114]	1.03	0.93	0.99	0.84	0.81	0.82	0.80	0.76	0.76	0.72	0.71	0.78	0.78	0.71	0.75	0.76	0.81	
D2-Net [39]	2.13	1.56	1.70	1.65	1.65	1.55	1.39	1.38	1.45	1.35	1.28	1.41	1.52	1.21	1.26	1.34	1.49	
KP2D [139]	1.08	0.91	1.05	0.99	0.96	0.98	0.92	0.84	0.93	0.85	0.79	0.87	0.90	0.83	0.83	0.88	0.91	
SiLK [50]	1.11	0.88	0.99	0.96	0.96	0.93	0.83	0.83	0.86	0.82	0.76	0.86	0.90	0.77	0.83	0.83	0.88	
Cross-domain																		
SuperPoint [34]	13.94	9.51	9.70	14.02	15.08	10.37	35.37	14.86	13.65	2.68	3.02	3.02	2.48	4.13	2.22	2.38	9.78	
R2D2 [114]	20.02	12.44	13.29	17.95	19.33	11.40	50.52	18.06	18.79	2.63	3.19	3.56	2.27	4.64	2.07	2.22	12.65	
D2-Net [39]	14.99	10.54	10.23	13.66	13.63	8.86	32.66	13.02	12.26	3.99	4.03	4.32	4.13	4.93	3.35	3.60	9.89	
KP2D [139]	19.90	13.34	14.37	16.58	16.11	11.36	46.94	16.70	16.52	3.05	3.94	3.62	2.57	5.63	2.40	2.65	12.23	
SiLK [50]	19.76	12.03	14.09	33.15	33.36	19.46	60.70	24.80	26.54	3.68	5.08	5.39	3.15	8.78	2.88	3.53	17.84	

Relative pose errors. In some cases, the sampled keyframe pairs have large relative poses, leading to significant occlusion, especially across different trajectories, making accurate relative localization unfeasible. To account for such outliers, we compute the median pose errors for each trajectory pair. In Tab. 5.4, we report metrics for both intra-domain and cross-domain cases. In intra-domain localization, relative poses are estimated for image pairs captured along the same trajectory (same visual conditions), whereas in cross-domain localization, they are estimated for different trajectories (different visual conditions). In the cross-domain case, the reported errors per reference trajectory are the mean of all median pose errors between the reference and all query trajectories. Furthermore, we plot in Fig. 5.5 the percentage of correctly localized images under different thresholds across all trajectory pairs. While this does not provide insight into localization performance along individual trajectories, these plots offer a better understanding of the distribution of localization errors.

In Tab. 5.4, all feature extraction networks exhibit significantly lower localization errors in intra-domain localization. Furthermore, this performance gap is clearly illustrated in Fig. 5.5. In intra-domain localization, R2D2 [114] consistently outperforms other methods, while D2-Net [39] struggles. In contrast, D2-Net demonstrates robust results in cross-domain localization for scenes with larger variations, outperforming R2D2, KP2D [139] and SiLK [50]. SiLK, in particular, performs poorly in cross-domain localization, consistently yielding the worst results overall and significantly underperforming in the Office Loop. The results presented in the table suggest that trajectories BC1 and OL4, both characterized by sunny conditions, pose particularly challenging localization tasks within their respective scenes, likely due to the presence of shadows. These shadows introduce edges in one domain that are absent in the other, complicating cross-trajectory localization. The challenges in cross-domain localization also

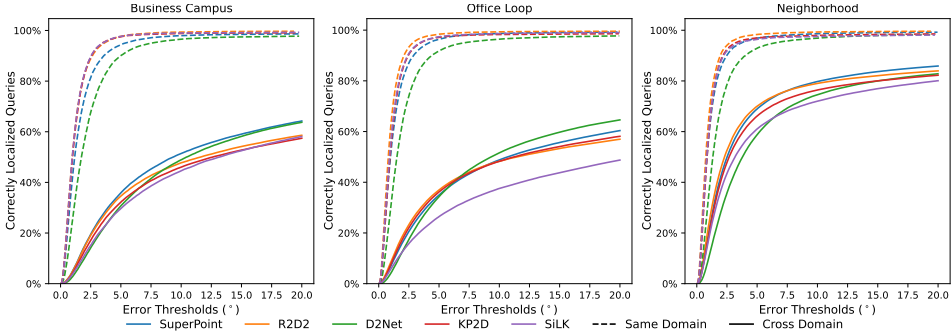


Figure 5.5: Cumulative Distribution of Relative Pose Errors on the 4Seasons Dataset [156]. For each scene, we present the AUC curves separately evaluated against our ground truth poses. Cross-domain plots are marked with filled lines while intra-domain plots are marked with dashed lines. The plots reveal a large performance gap between intra- and cross-domain localization.

escalate under snowy conditions (as seen in BC3 and OL5), given that snow introduces markedly different textures and leafless trees present new geometries.

5.4.2 Benchmarking Absolute Pose Estimation

We further investigate the cross-domain localization performances of the extractors by calculating the absolute 6 DoF pose of query images with respect to a reference map, utilizing the HLoc localization approach [117]. Building on the poses available in our maps, the reference maps are reconstructed using local features extracted by the feature extractor being evaluated. For a given query image, image retrieval is performed to obtain the n most similar images from the database of the reference images. This process effectively narrows down the search space by focusing on the most relevant matches rather than searching throughout the entire 3D model.

In Tabs. 5.5 and 5.6, following [124], we report the percentage of localized images under three thresholds: (0.25m, 2°), (0.5m, 5°) and (5m, 10°). We report the results using two methods of image retrieval. The first method (Tab. 5.5) utilizes the ground-truth poses of the images to retrieve the closest reference images to the query. The second method (Tab. 5.6) relies on NetVLAD [6]. We compute the localization accuracy per reference-query trajectory pair. For each scene, we present cross-domain localization performances, averaging accuracies over all trajectory pairs.

R2D2 and SuperPoint achieve the highest number of correctly localized images under the finer thresholds (0.25m/ 2° , 0.5m/ 5°) while D2Net reports the best localization result under the coarse threshold (5m/ 10°). R2D2 [114] performs better than SuperPoint [34] in scenes with smaller domain variations. Conversely, SuperPoint [34] demonstrates greater robustness to larger variations in viewing conditions that are encountered in the Business Campus and Office Loop scenes.

The results that are obtained using NetVLAD for image retrieval mirror those obtained by using the ground truth poses except that the values are lower. This outcome is expected as using ground-truth pose for image retrieval facilitates the retrieval of the query images closest to the reference image. Utilizing ground-truth poses in this manner helps in isolating the localization errors from image retrieval errors, with the former stemming from the quality of the reconstructed map, as well as the description and matching of local features. Similarly to the case of relative pose estimation, SiLK struggles in cross-domain localization. KP2D was omitted from this experiment, as it was not implemented in HLoc.

Table 5.5: Cross-domain absolute pose accuracies using ground-truth poses for image retrieval on the 4Seasons dataset [156]. For each scene, the percentages of accurately localized query images under thresholds (0.25m, 2°), (0.5m, 5°), and (5m, 10°) are evaluated and reported against our ground truth poses.

Retrieval Scenes	Ground-Truth		
	Business Campus	Office Loop	Neighborhood
SuperPoint [34]	66.6 / 82.6 / 99.7	47.2 / 60.7 / 89.9	71.5 / 82.0 / 97.6
R2D2 [114]	62.1 / 80.7 / 99.8	43.2 / 58.06 / 88.3	72.2 / 83.4 / 98.0
D2-Net [39]	61.8 / 83.1 / 99.9	42.9 / 59.1 / 92.3	65.8 / 80.1 / 98.5
SiLK [50]	44.9 / 61.2 / 85.4	36.0 / 46.6 / 68.0	65.5 / 77.7 / 93.7

Table 5.6: Cross-domain absolute pose accuracies using NetVLAD [6] for image retrieval on the 4Seasons dataset [156]. For each scene, the percentages of accurately localized query images under thresholds (0.25m, 2°), (0.5m, 5°), and (5m, 10°) are evaluated and reported against our ground truth poses.

Retrieval Scenes	NetVLAD		
	Business Campus	Office Loop	Neighborhood
SuperPoint [34]	64.5 / 80.3 / 97.3	44.4 / 56.1 / 78.6	70.5 / 80.6 / 95.3
R2D2 [114]	60.2 / 78.6 / 98.2	41.3 / 54.7 / 79.9	71.4 / 81.9 / 96.2
D2-Net [39]	59.7 / 80.9 / 98.6	40.6 / 55.3 / 83.1	64.8 / 80.0 / 96.5
SiLK [50]	42.2 / 57.4 / 80.2	33.0 / 42.3 / 59.7	64.3 / 76.0 / 90.7

In all experiments, a noticeable performance gap persists between intra-domain and cross-domain localization across all feature extractor methods, regardless of the domain type. Although these extractors demonstrate excellent localization performance in intra-domain tasks, the persistent gap highlights that significant improvement is needed to achieve descriptors robust to long-term illumination variations. Current state-of-the-art research has made substantial progress in refining extractors through model-centric approaches, such as enhancing models and optimization schemes. However, we argue that these approaches are insufficient for closing this gap because they do not account for real cross-domain variations across correspondences. Instead, we advocate for a shift towards data-centric approaches. In the forthcoming sections, we will explain the need for real cross-domain correspondences to supervise feature extractors and enforce cross-domain invariance. Furthermore, we present iCDC in detail

and demonstrate a significant reduction in the performance gap when using its correspondences for supervision.

5.5 Network Supervision for Long-Term Invariant Descriptors

We can reduce the performance gap observed in the previous section by supervising feature extractors with real cross-domain correspondences. However, manually annotating correspondences is impractical, as it would require per-pixel annotations for each image pair. As a result, state-of-the-art methods [114, 34, 139, 50] employ homographic transformations to flexibly generate correspondences between the reference and transformed images. However, homographies cannot encapsulate real-world geometric variations. Here, image augmentations are necessary to generalize across lighting variations.

5.5.1 Correspondence Generation Overview

Structure-based methods provide a reliable approach to generating correspondences across different views [3, 76]. However, relying solely on sparse SfM maps results in a sparse set of correspondences. The initial sparse reconstruction from SfM can be further processed with Multi-view stereo to obtain a denser reconstruction. However, the resulting depth is often noisy and yields suboptimal correspondences.

Dense correspondence estimation is a field where networks estimate dense matches between image pairs and could be used to generate dense cross-domain correspondences for supervision [114]. However, estimating dense correspondences across long-term viewing conditions is particularly challenging due to significant lighting, texture, and geometry variations.

In Fig. 5.6, we present an overview of all correspondence generation methods we explore. In addition to homographies, we use a dense correspondence network, WarpC [141], to estimate dense correspondences across views and our refined maps to generate sparse correspondences. WarpC, trained using an unsupervised method through warp consistency, demonstrates state-of-the-art performance and robustness to varying views and domains across image pairs. Our refined maps provide highly accurate relative poses and per-keypoint depth values. Although direct use of the matches stored in the maps is possible for correspondence generation, we found the resulting set too sparse and lacking in variation. By reprojecting the keypoints across arbitrary views, we obtained more varied correspondences and better supervision despite encountering significant occlusion. Lastly, the figure includes iCDC, our method, which is detailed in the following subsection.

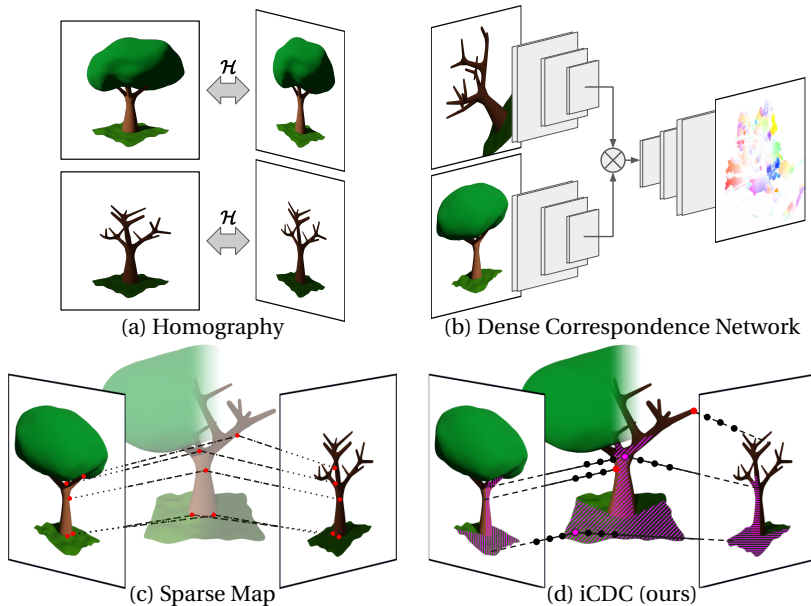


Figure 5.6: Overview of explored correspondence methods. Homographic transformations (a) are used to train most extractor methods out of convenience. Dense correspondence estimation networks (b) can estimate correspondences between real-world images across arbitrary viewpoint changes. However, these methods can be inaccurate, particularly across visual domains. Sparse SfM map matches (c) can serve as correspondences, providing highly accurate sparse correspondences across domains and views. In our method iCDC (d), we train domain-specific NeRFs using cross-trajectory SfM maps and generate correspondences by comparing the implicit 3D geometries. Highlighted regions are where correspondences are extracted.

5.5.2 iCDC: Implicit Cross-Domain Correspondences

Generating correspondences with NeRFs. A NeRF [97] learns an implicit continuous 3D representation of a scene from a collection of images with known camera viewpoints. In the computer graphics community, NeRFs have been used to synthesize novel photorealistic views of a scene. The computer vision community has also adopted NeRFs, leveraging the power of these implicit representations in various domains [92, 60]. NeRFs show great promise in the field of scene understanding [150], and they have the potential to revolutionize SLAM algorithms [175, 174]. Beyond synthesizing RGB data, explicit 3D representations of the implicit scenes can be rendered. NeRF-supervision [166] generates correspondences across image pairs by rendering corresponding depth images and comparing the underlying 3D geometries. In a controlled environment with consistent lighting, the authors demonstrated how a NeRF trained in such conditions could generate correspondences across images of an object by reprojecting rendered depth maps from one frame to another. They showcased the

powerful cross-view information aggregation abilities of NeRFs by accurately generating correspondences across reflective objects – a canonical challenge.

iCDC overview. Building on NeRF-supervision [166], we introduce iCDC, capable of generating accurate correspondences across images captured in different visual domains. iCDC leverages the fact that a scene’s underlying static 3D geometry, such as buildings or street signs, remains unchanged, regardless of visual domain changes, such as daytime, weather, and seasonal variations. For each scene, a single NeRF is trained per trajectory, i.e., per visual domain, using the poses from our multi-trajectory refined maps.

Training a single NeRF using data from different trajectories with varying visual domains results in an inferior scene representation. NeRF-W [92] addresses this issue by learning per-image latent appearance codes, which makes NeRFs more robust to lighting variations. However, modeling long-term variations remains far more challenging. For example, consider a tree present in one trajectory but cut down in another. If a NeRF were trained on both trajectories, it would produce a scene representation that cannot be accurate for both cases. Consequently, relying on the NeRF’s underlying 3D representation would lead to false positive correspondences of the tree.

In iCDC, although the per-trajectory NeRFs do not share training data, they are expressed within the same coordinate system. Thanks to the high accuracy of the maps, the implicit 3D representations of the NeRFs share the same space and can be compared to generate real cross-domain correspondences. Specifically, we first render depth maps using the respective NeRFs for a pair of images captured along different trajectories. We then reproject the points using the corresponding map poses.

Scaling NeRFs to represent 4Seasons scenes. The 4Seasons dataset comprises long trajectories that span large areas, making it challenging to represent entire scenes with a single NeRF. Following recent approaches [142, 136], we evenly cluster the scenes into blocks based on frame poses and train a small NeRF for each block. Since 4Seasons trajectories can revisit the same location multiple times, a single NeRF block can contain multiple segments of the same trajectory. Near the edges of these blocks, rendering quality suffers due to the limited number of observations available for training the NeRFs. To mitigate this, blocks are expanded with buffer regions that overlap with adjacent blocks. Frames in these buffer regions are used exclusively for training, allowing the blocks to share training data while ensuring each frame has a designated NeRF for rendering depth maps. For convenience, we refer to the group of NeRFs representing an individual trajectory as a trajectory-NeRF. During testing, a trajectory-NeRF functions just like a regular NeRF, rendering frames at the input camera pose locations.

Supervising NeRFs with sparse depth. DS-NeRF [33] demonstrated that supervising a NeRF with sparse depth values can significantly enhance rendering performance with fewer training views. In our scenario, we found that sparse depth supervision is both beneficial and necessary for rendering accurate depth maps. This necessity arises because the 4Seasons data is captured using a forward-facing stereo camera, offering minimal view variation. Additionally, the scenes are unbounded, meaning that scene

points lie at large distance variations from the camera. Moreover, the environments are uncontrolled, leading to drastic changes in lighting conditions along a trajectory. Each of these factors makes interpreting depth from only RGB images a significant challenge. The additional depth supervision provides sufficient information for the NeRFs to effectively learn the underlying 3D representations. Following the DS-NeRF approach, we utilize sparse depth supervision to enhance the depth rendering results by extracting sparse depth values per frame directly from our sparse SfM maps.

Depth-frame rendering. NeRF-Supervision [166] proposed to treat the rays as depth probability distributions, sampling depth values during run-time. Instead, we follow the Instant-ngp [99] implementation. By modeling the transmittance along a ray, we can render depth by estimating the first instance after the light is absorbed by a surface in the scene. Furthermore, we can generate depth data prior to extractor training instead of having to sample depths during the training.

Correspondence generation. For a pair of images $\mathbf{I} \in \mathcal{U}$ and $\mathbf{J} \in \mathcal{V}$, belonging to trajectories \mathcal{U} and \mathcal{V} respectively. We render depth maps using trajectory-NeRF- \mathcal{U} and trajectory-NeRF- \mathcal{V} . Then, we generate candidate correspondences across the images by reprojecting the depth map of \mathbf{I} onto \mathbf{J} using the poses stored in the map. In the case where $\mathcal{U} = \mathcal{V}$, the same NeRF will render the depths for both frames.

We use the loop consistency test, similar to WarpC [141], to filter out invalid correspondences. The candidate pixel of \mathbf{J} corresponding to a reference pixel of \mathbf{I} has its own rendered depth value. We reproject this depth value back onto image \mathbf{I} , arriving at a certain 2D loop consistency distance from the reference pixel. We filter out correspondences with loop consistency distances that are higher than a user-defined loop consistency threshold α .

While the loop consistency test removes most invalid correspondences, there can be corner cases with false positives. For instance, if the relative pose between the reference and query frame involves only a transformation along the frame normal (as is common with many frame pairs in the 4Seasons dataset), the center pixels will always pass the loop consistency test. To address this, we compare the reprojected depth value of a reference pixel with the depth value of the corresponding query pixel. The absolute difference between these depth values is termed the depth consistency difference. Correspondences with depth consistency differences greater than a user-defined threshold (denoted as β) are filtered out.

We present in Fig. 5.7 examples of generated correspondences. In the first example, we filter using only the loop consistency test, while in the second example, we filter using both the loop and the depth consistency tests.

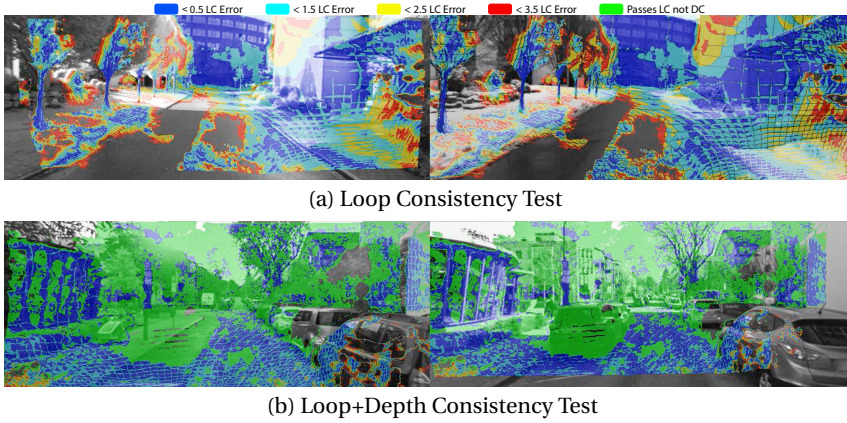


Figure 5.7: The figure presents image pairs illustrating iCDC Correspondences. Fig. (a) shows pixels of correspondences satisfying loop consistency (LC) tests colored according to LC thresholds. Fig. (b) depicts the same mask, except marking pixels as green that pass the LC test but not the depth consistency (DC) test with a 15cm DC error threshold.

5.6 Experimental Setup Details

5.6.1 Train-test-split

We train on two scenes from the 4Seasons dataset, Office Loop and Neighborhood, and test on the third scene, Business Campus. In both the training and testing splits, we use spatiotemporal keyframing to reduce image pairs with minor variations. For the training split, consecutive reference frames are keyframed approximately 6 meters apart along a single trajectory loop. For each reference frame, we sample up to two spaced-out query frames per trajectory loop, with a maximum distance of 6 meters from the reference. If the trajectory loops around, the number of reference keyframes and query frames per reference are both multiplied by the number of loops. This approach results in 16,266 intra-domain and 77,428 cross-domain training pairs. The keyframing methodology for the testing data is explained for each experiment below.

5.6.2 Constructed Maps

We outlined our approach in Sec. 5.3 for constructing joint-trajectory maps to generate accurate relative poses across domains. Since we only compare image pairs, we require relative poses across two trajectories at a time. Therefore, we constructed maps for all trajectory pairs for a given scene. For each trajectory pair, we construct two types of maps. For benchmarking local features, we construct maps from a subset of the frames, those with registered geolocations. For NeRF training, more frames result in better performance, so we generated a second set of maps using all available frames,

as described in Sec. 5.3.

5.6.3 iCDC Setup

NeRF training. Although our multi-trajectory maps were constructed using images captured by a single camera, the 4Seasons dataset provides stereo image pairs. To train the NeRFs, we utilize images from both cameras. For efficient NeRF training, we employ Instant-ngp, which is architected for fast processing, ensuring quick training and rendering essential for our block-wise NeRF approach. The scene is divided so that the farthest apart frames in each block are approximately 64 meters apart. Additional details about the engineering solutions for representing unbounded outdoor scenes using Instant-ngp are provided in the supplementary material.

Generating correspondences. We generate correspondences across an image pair using the corresponding rendered depth maps. Subsequently, we filter out invalid correspondences through loop consistency and depth consistency tests, as detailed in Section 5.5. For the loop consistency test, we set a threshold of $\alpha = 2$ (in pixel units). For the depth consistency test, anchoring poses using geolocations allowed us to use metric units. We observed a significant performance improvement with the implementation of the depth consistency test, selecting a threshold of $\beta = 15$ cm.

5.6.4 Extractor Training Setup

While our cross-domain correspondences can supervise any feature extractor, we chose R2D2 [114] because of its unique training loss characteristics. Besides optimizing for keypoint repeatability, which includes “peakiness” and “cosimilarity” losses, it also optimizes for descriptor reliability. We discovered that the repeatability loss needed adaptation for cross-domain correspondence supervision. Originally designed for homography-based correspondences with no significant geometric variations across matched keypoints, the loss function required modifications to accommodate cross-domain correspondences.

To train R2D2, we employed cosimilarity loss supervision exclusively for intra-domain training pairs for all cross-domain correspondence methods. More details and insights about the losses are provided in the supplementary.

To demonstrate the importance of real cross-domain correspondences, we trained an additional extractor to indicate that the improvements were not network-specific. We train SiLK [50] – we selected it due to the simplicity of its training pipeline.

We trained both networks using their default pipelines, making only the necessary adjustments to accommodate our correspondences. For R2D2, we updated the loss functions, while for SiLK, we modified the pipeline to support non-homography-based correspondences.

In both the R2D2 and SiLK training pipelines, images are augmented using noise, random cropping and scaling. Additionally, when supervising with real correspondences, we apply homographic transformations as a form of data augmentation. The SiLK pipeline only supports homography-based supervision. To enable SiLK to utilize real correspondences, we incorporate functions from R2D2. Specifically, we use a function to map ground truth correspondences across an image pair after applying a perspective homographic transformation to the query image.

To expedite training, we initialize both R2D2 and SiLK with their default weights. However, we use a sufficiently large learning rate to ensure that previous patterns are mostly overwritten. Using default weights for initialization provides only a marginal improvement over starting with random weights.

5.7 Results

5.7.1 Overview of Experiments

We train R2D2 using multiple correspondence methods on the 4Seasons dataset and observe a significant reduction in the performance gap when utilizing real cross-domain correspondences. We establish two baselines for our networks to ensure that the performance improvements are not solely due to the training data. Firstly, we use the default weights as a baseline. Secondly, we create another baseline by training R2D2 using homographies (its default pipeline) on the 4Seasons dataset. In our first experiment, we use two straightforward real cross-domain correspondence generation methods for training. We introduce iCDC to enhance the accuracy of these cross-domain correspondences. In a second experiment, we benchmark the iCDC-supervised extractor by generating correspondences using different methods and training our feature extraction and description backbone with them.

We summarize the correspondence generation methods:

- **Homography:** We utilize the default R2D2 training scheme, where individual images are loaded, a perspective homography is applied, and the resulting image pairs are then individually augmented.
- **Dense Correspondence Network:** We use WarpC [141] to obtain dense correspondences between an image pair. A loop consistency test is used to filter out inaccuracies. Both images are individually augmented, one of which is also augmented using homographic transformation. This same augmentation strategy was also implemented for the following two methods.
- **Sparse SfM structure:** We extract sparse depth values from our maps and reproject them across image pairs to generate correspondences.
- **iCDC:** We generate correspondences through iCDC, using training image pair

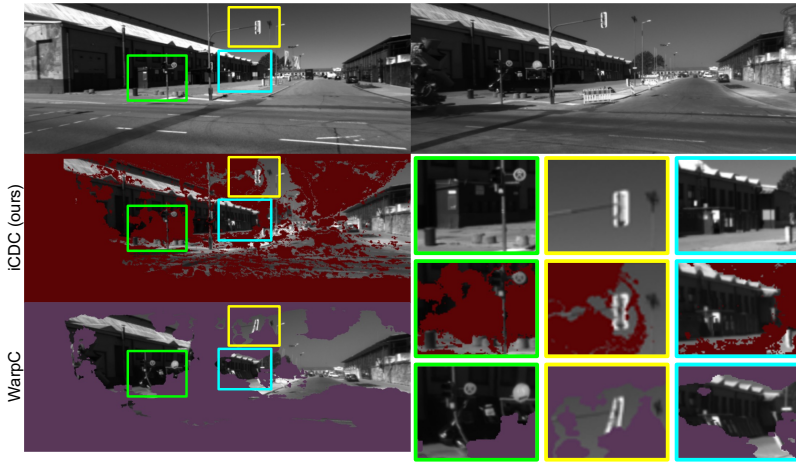


Figure 5.8: Comparison between our NeRF correspondences and WarpC [141] correspondences. Correspondences are generated for an image pair (top row). The second and third rows of the left column show WarpC and iCDC correspondences, respectively. More specifically, the right-hand side image is warped onto the left-hand side image following the correspondences. We zoom in on windows for detailed examination. In the case of accurate correspondences, the warped frames align with the left-hand side frame.

and depth maps rendered by corresponding trajectory-NeRFs. Candidate correspondences are filtered using loop and depth consistency tests.

5.7.2 Qualitative Correspondence Analysis

In Fig. 5.8, we evaluate the correspondence quality between the dense correspondence estimator WarpC and iCDC. To ensure a fair assessment, we employ only the loop consistency test to filter out candidate correspondences (marked in red and purple) for both methods. We also adjust the thresholds for both methods to yield a comparable number of correspondences per image pair. We generate correspondences for a pair of images (top row) using both methods. Using these correspondences, we warp the right-hand side image to align with the left-hand side image. The results of this alignment are displayed in the second row (iCDC) and third row (WarpC), with close-up views of selected regions for detailed comparison.

For both methods, correspondence errors are often difficult to detect visually; however, WarpC experienced significantly more frequent failures. In the zoomed-in regions, the frames generated by the warped dense correspondence estimator are notably misaligned.

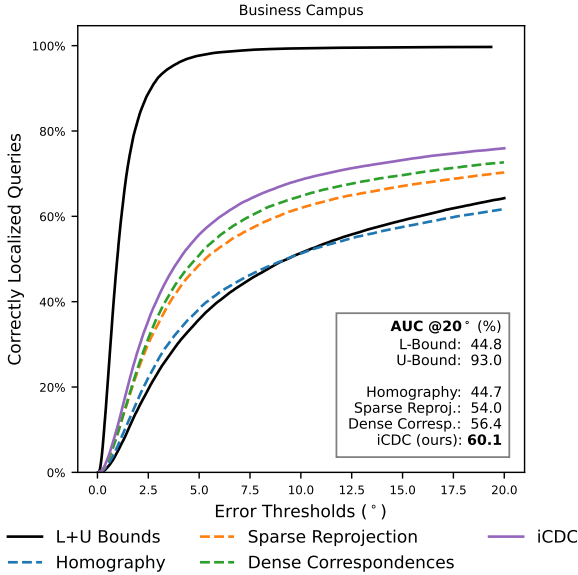


Figure 5.9: Cumulative distribution of cross-domain pose errors on the 4Seasons dataset [156] by supervising R2D2. Our trained R2D2 extractors are benchmarked using our ground truth poses (a). Lower and upper bounds (L+U Bounds) are the error distributions of the best cross-domain localization method (SuperPoint [34]) and R2D2 [114] evaluated on intra-domain respectively). Dashed lines are our trained extractors used to benchmark iCDC.

5.7.3 Relative Pose Estimation on Our Benchmark

For relative pose estimation, we utilize the benchmarking procedure outlined in section 5.4, focusing exclusively on Business Campus trajectories. We select reference frames every 10 meters along the trajectories. Query frames are chosen every two meters, not exceeding eight meters from the reference frame. Using extractors, we estimate the relative poses between image pairs, and we evaluate the performance using the ground truth poses available in our maps. To thoroughly analyze the error distributions in cross-domain localization, we present the cumulative error curves in figures 5.9 and 5.10.

Discussing primary extractor results (R2D2). In Fig. 5.9, we set SuperPoint with default weights as the lower bound – the best-performing pre-trained extractor on cross-domain localization. As an upper bound, we set the intra-domain localization performance of R2D2 with default weights.

Using the default homography pipeline to train R2D2 shows only a marginal improvement in cross-domain localization performance on our benchmark, and it is still surpassed by SuperPoint with default weights. This suggests that the slight gains are not due to the 4Seasons training images. However, employing supervision with cross-

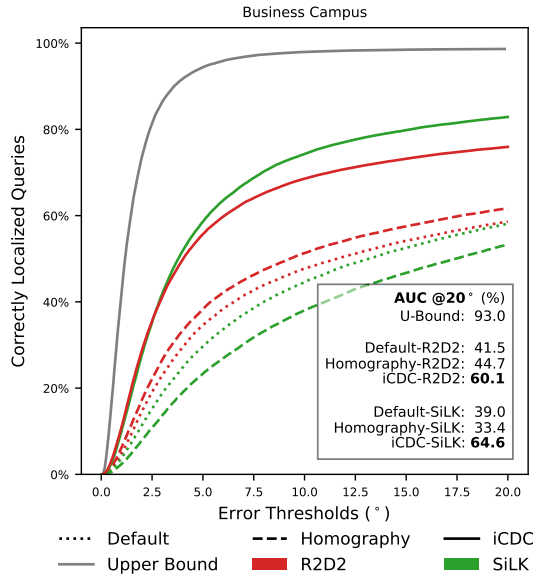


Figure 5.10: Cumulative distribution of cross-domain pose errors on the 4Seasons dataset [156] by applying our supervision method on SiLK. We complement Fig. 5.9 by plotting the results of our trained SiLK method. Upper bound results are obtained from the best cross-domain localization method (SuperPoint [34]) and R2D2 [114] evaluated on intra-domain, respectively). Dashed lines are our trained extractors used to benchmark iCDC.

domain correspondences consistently achieves a reduction in median error by at least 50% in cross-domain localization. There is a significant performance discrepancy between the homography-based method and our least effective extractor supervised with cross-domain correspondences. This substantial increase in performance underscores the importance of data-centric strategies. Even naive approaches, such as supervising with sparse map-based correspondences or using outputs from dense correspondence networks, significantly enhance robustness to long-term environmental changes.

Despite the high accuracy of the correspondences extracted from the sparse maps, using the estimates from the dense correspondence network yields better results. Although the accuracy of correspondences is crucial, the high variation of positive anchors during training provided by the dense correspondences was also an important factor. Through our qualitative analysis, we identified that generating correspondences using our iCDC network results in higher accuracy dense correspondences compared to the dense correspondence estimator WarpC. We achieved the best results when supervising R2D2 using iCDC correspondences due to the increased density and improved accuracy of the correspondences. Referring to the AUC of the pose errors at a threshold of 20°, we improved the cross-domain localization performance of R2D2 by 44.8% and reduced the performance gap between the default R2D2 and R2D2 trained using our iCDC method by 36.1%.

Extending discussion to SiLK. To ensure that the findings from this study are not specific to R2D2, we expanded our training to include SiLK, using a smaller subset of the correspondence methods. For clarity, we have presented the results of SiLK supervision in a separate figure (Fig. 5.10). SiLK shows unique characteristics. As detailed in the benchmarking section (Sec. 5.4), SiLK achieves state-of-the-art results in same-domain localization but performs poorly in cross-domain localization. Our experiments suggest that SiLK tends to overfit the training data more than R2D2, and if the training data lacks sufficient variety or does not visually resemble the test domain, SiLK struggles to generalize effectively.

We trained using the 4Seasons dataset with the default SiLK training pipeline (homography). Contrary to what was observed with R2D2, SiLK trained this way performs worse than when using default weights. The network overfits to the training scenes, and since the 4Seasons dataset is less varied than the default training data for SiLK, this results in the poorest cross-domain performance observed in our study. Interestingly, when SiLK is supervised using iCDC correspondences, it achieves the highest performance across all our tested networks. The aggressive fitting of cross-domain correspondences proves highly effective on our test scene despite the limited training image variation.

5.7.4 Absolute Pose Estimation on Our Benchmark

We localize a query image using feature extractors with respect to a reference 3D map. Our approach follows the benchmark setup for absolute pose estimation as previously detailed in sec. 5.4. For each trajectory in the testing scene, we triangulate an SfM map using an extractor and ground truth poses. For image retrieval, we utilize NetVLAD [6]. The localization results under thresholds (0.25m, 2°), (0.5m, 5°), and (5m, 10°) are reported in Tab. 5.7. Metrics are averaged per reference trajectory and are also globally averaged.

We observe similar patterns as in the relative pose estimation case, both for R2D2 and SiLK. Supervising R2D2 using the default pipeline increases the cross-domain localization performance, however, does not outperform SuperPoint with default weights. Supervising R2D2 using cross-domain correspondences uniformly outperforms the default extractors. Moreover, the accuracy of our iCDC correspondences yields the best results. However, the reference maps contain a very dense set of frames. Consecutive frames are no more than 30 cm apart, and the Business Campus trajectories loop around three times. Therefore, the task of NetVLAD is made easy, and retrieved reference and query frames have very small relative poses, simplifying the localization task. We find that there is less room for improvement than in the relative pose estimation case, with the performance increases being less dramatic. In relative pose estimation, we engineer a broader range of relative poses between test image pairs.

Table 5.7: Cross-domain absolute pose accuracies on the 4Seasons dataset [156]. For our test scene, the percentages of accurately localized query images under thresholds (0.25m, 2°), (0.5m, 5°), and (5m, 10°) are evaluated and reported against our ground truth poses. Results using default weights, our trained R2D2 networks, and our trained SiLK networks are grouped separately.

Traj.	BC1	BC2	BC3	ALL
Pretrained Benchmarks				
SuperPoint [34]	61.4/78.1/96.3	67.1/79.7/98.4	65.1/81.9/97.6	64.5/79.9/97.4
R2D2 [114]	55.0/75.2/97.6	63.6/79.3/99.1	61.9/81.2/98.0	60.2/78.5/98.2
D2-Net [39]	57.4/79.6/98.0	60.7/80.5/99.2	60.9/82.4/98.6	59.7/80.9/98.6
SiLK [50]	34.0/46.9/70.1	50.2/64.9/88.3	42.5/60.5/82.2	42.2/57.4/80.2
Supervising R2D2 [114]				
Homography	61.6/77.5/97.9	67.3/80.1/99.3	64.4/82.6/98.2	64.4/80.1/98.5
Sparse Map	64.2/79.2/ 98.4	70.7 /81.1/99.5	67.5/84.0/98.5	67.5/81.4/98.8
Dense Corresp.	64.4/ 80.8 /98.3	67.4/ 82.0 /99.5	66.8/ 85.4 /98.7	66.2/82.7/98.8
iCDC	66.3 /80.6/ 98.4	69.1/81.8/ 99.6	69.0 /85.3/ 98.8	68.2/82.6/98.9
Supervising SiLK [50]				
Homography	25.7/38.9/64.1	42.3/58.4/83.1	41.7/59.0/81.1	36.6/52.1/76.1
iCDC	63.0/77.2/97.1	69.2/80.1/98.5	67.4/83.7/98.0	66.5/80.3/97.8

5.7.5 Evaluation of Our Correspondences on the Visual Localization Benchmark

To investigate the generalizability of our feature extractors supervised by iCDC, we evaluate their localization performances in the long-term localization benchmark [124]. The benchmark datasets are captured using different camera setups, each presenting its own challenges.

Overview of the benchmark datasets. The CMU-Seasons [8] setup is the most similar to our own, featuring a camera mounted on a car capturing scenes that span both rural and urban environments, with car trajectories that traverse various visual domains. However, there are notable differences: the cameras in the 4Seasons dataset are forward-facing, whereas those in the CMU dataset are rotated 45°. Additionally, the CMU dataset includes a substantially larger proportion of vegetation-focused scenes compared to the 4Seasons dataset.

The RobotCar-Seasons dataset [90] features images captured by side-facing and rear-facing cameras, the latter of which are more akin to the setup used in the 4Seasons scenario. Notably, all query frames in the benchmark are from rear-view images. Moreover, RobotCar is captured in an urban setting, which offers more distinct and recognizable geometries than those found in the CMU dataset. In addition to visual domain variations similar to 4Seasons, a primary benchmark of RobotCar includes the challenge of localizing night-time images against a day-time reference map, a scenario not covered in our dataset since we do not have night-time images. Moreover, the image quality is significantly lower than that of 4Seasons, particularly the night-time im-

ages with strong motion blur.

The Aachen Day-Night dataset [125] also focuses on night-time image localization, with the image quality being significantly higher compared to other datasets. However, Aachen Day-Night poses the greatest challenge for our extractors. It features a hand-held camera setup, leading to substantially different relative views between images. Whereas the 4Seasons dataset primarily captures views along roads with buildings on the sides, the Aachen Day-Night dataset predominantly features direct views of buildings.

Localization results, using NetVLAD for image retrieval, on the long-term visual localization benchmark are listed in tables 5.8, 5.9, and 5.10.

Table 5.8: Benchmarking long-term visual localization on CMU Seasons Extended dataset. We evaluate the performance of our extractors on the long-term localization benchmark [124] and compare them to the performances of SuperPoint, R2D2, and SiLK with default weights.

Scenes	CMU Seasons Extended [8]		
	Traj.	Urban	Suburban
SuperPoint [34]	93.5 / 96.1 / 98.2	84.2 / 86.9 / 92.9	78.2 / 81.9 / 85.8
R2D2 [114]	94.4 / 97.3 / 99.0	85.3 / 88.9 / 93.5	74.2 / 78.8 / 83.6
iCDC-R2D2	96.5 / 98.8 / 99.4	93.4 / 94.9 / 97.1	87.5 / 90.3 / 92.9
<hr/>			
SiLK [50]	57.6 / 62.0 / 68.0	42.5 / 47.9 / 57.6	17.0 / 19.6 / 26.6
iCDC-SiLK	92.2 / 94.9 / 96.9	81.8 / 84.9 / 89.2	63.2 / 67.2 / 72.4

Table 5.9: Benchmarking long-term visual localization on RobotCar seasons dataset. We evaluate the performance of our extractors on the long-term localization benchmark [124] and compare them to the performances of SuperPoint, R2D2, and SiLK with default weights.

Scenes	RobotCar-Seasons v2 [90]	
Traj.	Day All	Night All
SuperPoint [34]	64.2 / 93.7 / 99.1	14.0 / 27.7 / 33.8
R2D2 [114]	64.5 / 94.4 / 99.2	16.8 / 32.6 / 39.4
iCDC-R2D2	64.9 / 94.6 / 99.4	24.5 / 45.5 / 52.2
SiLK [50]	62.2 / 90.8 / 96.1	12.6 / 18.4 / 21.7
iCDC-SiLK	64.5 / 94.0 / 99.2	15.4 / 26.8 / 34.3

Discussing primary extractor results (R2D2).

On the driving datasets, iCDC-R2D2 correspondences have outperformed the default SuperPoint and R2D2 across all metrics. Notably, in the CMU Seasons dataset, there are substantial performance boosts in both the Suburban and Park scenes. In Urban scenes, although the improvement is less pronounced, it remains significant given the already high performances. For day-time localization in the RobotCar-Seasons,

Table 5.10: Benchmarking long-term visual localization on Aachen Day-Night dataset. We evaluate the performance of our extractors on the long-term localization benchmark [124] and compare them to the performances of SuperPoint, R2D2, and SiLK with default weights. Aachen Day-Night presents the greatest challenge due to the large domain variation between hand-held and car-mounted camera setups.

Scenes	Aachen Day-Night v1.1 [125]	
	Traj.	
	Day	Night
SuperPoint [34]	87.1 / 93.3 / 96.7	68.1 / 80.6 / 90.1
R2D2 [114]	88.0 / 95.3 / 97.9	67.0 / 86.4 / 96.3
iCDC-R2D2	87.7 / 93.7 / 96.6	67.5 / 84.3 / 93.7
SiLK [50]	78.8 / 84.3 / 89.8	38.2 / 48.2 / 55.5
iCDC-SiLK	83.9 / 90.0 / 94.1	60.7 / 76.4 / 85.3

our extractor only marginally outperforms the baselines, potentially due to a NetVLAD bottleneck or the lower quality of images. However, during night-time localization, our extractor significantly outperforms the other baselines, suggesting that supervising domain variation improves performances in unseen domains. These results underscore the importance of improved data-centric approaches. Despite limited view variation in the 4Seasons dataset, we observe marked performance improvements in long-term localization scenarios. The real-world variations across correspondences even improve localization performances in unseen visual domains, such as in night-time localization.

Despite the significant shift in viewing conditions, our extractor delivers competitive performance on the Aachen Day-Night benchmark, surpassing SuperPoint in nearly all metrics. R2D2 with default weights recorded the best results in most metrics, but it's important to note that it was trained on day-time data from Aachen Day-Night as well as on day-to-night style-transferred versions of those images.

For detailed results on the CMU and RobotCar datasets, please refer to the supplementary material.

Extending the discussion to SiLK. Our benchmark revealed that SiLK aggressively fits the training data, struggles with generalizing, and achieves low cross-domain localization performance with default weights. Furthermore, training on 4Seasons with their default pipeline reduces the performance by overfitting. On the other hand, we found that supervising SiLK with iCDC correspondences yielded the best results on our benchmark. That being said, generalizing to the long-term localization benchmarks is a separate challenge. SiLK with default weights performs poorly on CMU and Aachen Day-Night, though it achieves results comparable to other baselines on RobotCar. Similarly to the case of our benchmark, iCDC supervision significantly improves the performance of SiLK on the long-term localization benchmarks by aggressively fitting the domain variations. This is only enough to make certain metrics comparable in CMU and Aachen. As RobotCar-Seasons' rear-view images present the scenario most similar to that of 4Seasons, iCDC-SiLK achieves competitive results.

5.8 Additional Insights

5.8.1 Training Learned Matchers Using iCDC

We also explored two related fields relevant to this work. First, NeRFs (Neural Radiance Fields) can be used for data generation. Instead of storing a dataset and sampling from a discrete set of frames, NeRFs allow for flexible sampling of images across a continuous range of camera poses. Additionally, views can be extrapolated beyond the original camera trajectory, creating new perspectives that could enhance the generalization of networks. Second, learned matchers [118, 86] rely on local features and learn matching patterns between image pairs. These matchers have significantly improved localization performance in complex environments, such as day-night localization in long-term localization benchmarks [124]. We investigated these two extensions of iCDC.

Training with novel views. We explore the impact of using rendered images in training, focusing on two key questions: (1) Can we maintain performance levels when training on rendered data? (2) Can we improve generalization to other datasets by using extrapolated views? To investigate these questions, we generated a secondary dataset rendered by our NeRFs. The images are rendered at the same X, Y, Z coordinates as the training data, but with incremental rotations around the gravity axis to the left and right. However, we found that extrapolation was challenging due to the forward-facing setup. We limited the rotation to $\pm \frac{\pi}{8}$, and even then, there was a noticeable decrease in rendering quality. For training, we used iCDC correspondences.

After training, we conducted evaluations using our benchmark. It is important to note that the extrapolated views will not improve performance on our benchmark, because such variations are not present. We measured the cross-domain performance; pose AUC in % under thresholds $5^\circ/10^\circ/20^\circ$ is: 28.1/43.1/55.7. The performance dips in comparison to our best network iCDC-R2D2 (31.5/47.4/60.1), but it did outperform all other feature extractors trained using homography (the best being 20.2/32.7/44.7), and while being outperformed, achieved comparable results to the best performing naive cross-domain correspondence generation approach, the dense correspondence network supervision: 28.1/43.6/56.4.

We present long-term localization benchmark results for many more extractors in Tabs. 5.11, 5.12, and 5.13. We get more balanced results for the training with synthesized views (Nov.-iCDC-R2D2+NN). Thanks to the improved generalizability, it performs approximately as well as iCDC-R2D2 and outperforms it in RobotCar and Aachen Day-Night in many metrics despite the suboptimal quality of the training data.

Learned matchers. LightGlue [86] introduces improvements in trainability, efficiency, and performance over the deep keypoint matcher SuperGlue [118]. We train a broad array of networks using the default setup but with two key modifications: (a) we introduce new types of data augmentation into the training pipeline, and (b) we train on the 4Seasons dataset using iCDC. The introduction of new augmentations (a) is crucial because we find that while feature extractors tend to overfit to texture patterns in images,

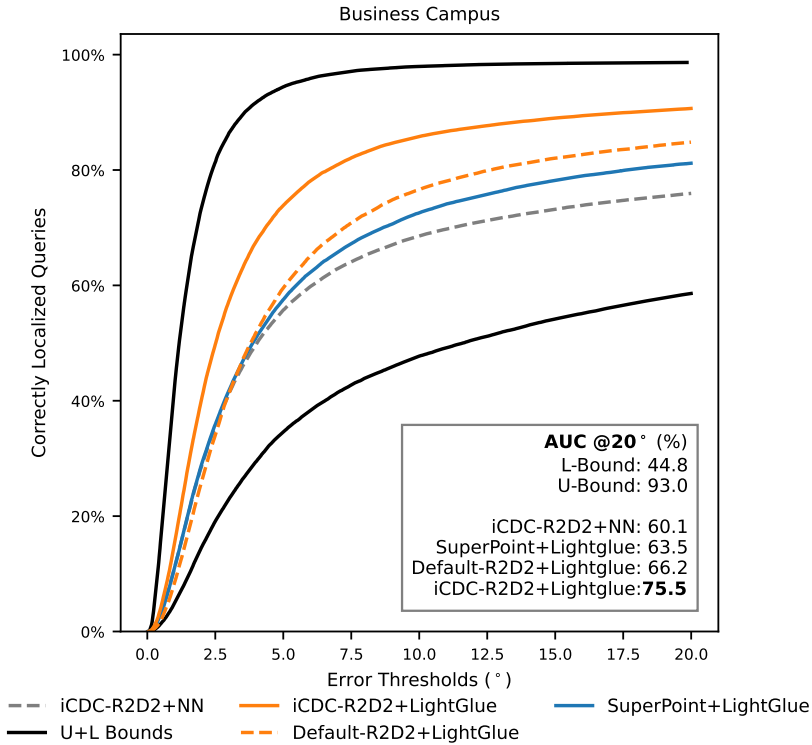


Figure 5.11: Cumulative distribution of cross-domain pose errors on our benchmark dataset. Keypoints are matched using either Euclidean distance-based Nearest Neighbors (NN), or using LightGlue [86]. The lower bound is the SuperPoint [34] with default weights. The upper bound is the intra-domain localization performance of [114] with default weights.

learned matchers are prone to overfitting repetitive matching patterns across image pairs. The original LightGlue training pipeline does not apply homographic transformations or cropping when training on Megadepth [82], due to the greater variety of images there compared to our case. In the 4Seasons dataset, it is very easy to overfit to matching patterns. For instance, keypoints on the left typically remain on the left in almost all cases, because the car moves forward in a straight line.

Although training strictly on the 4Seasons dataset leads to the best performances on our benchmark, the resulting matchers do not generalize at all to other datasets. To address this, we introduce a small amount of random cropping to explore the effect of overfitting to matching patterns. We present only these results.

Furthermore, in an attempt to generalize to other datasets, we also use our rendered extrapolated views for training. We present results for training LightGlue with these novel views exclusively on the long-term localization benchmark.

In Fig. 5.11, we present a small subset of the networks we trained, benchmarking against our extractors trained with iCDC using Euclidean distance-based nearest neighbors (NN) matching and against pretrained SuperPoint+LightGlue. As expected, matching with LightGlue led to a huge increase in performance over SuperPoint+NN, the lower bound in the plot. While SuperPoint+LightGlue outperforms our network with NN matching, iCDC-R2D2+NN, the results are comparable, highlighting the importance of cross-domain correspondence supervision.

To investigate the importance of descriptor quality when matching with learned matchers, we first trained LightGlue with default-R2D2 features. It marginally outperforms SuperPoint+LightGlue on our benchmark, however, to draw any conclusions about the networks or the training strategy, a further ablation study would be needed. Subsequently, we trained LightGlue using iCDC-R2D2. The resulting network and matcher achieved the greatest reduction of the performance gap with respect to default-R2D2, 66.0% when considering AUC @20°, and outperformed SuperPoint+LightGlue by 18.9%. On top of highlighting the power of learned matchers, the large gap between default-R2D2 and iCDC-R2D2 using LightGlue underscores the importance of data-centric research for training local features.

Table 5.11: Benchmarking long-term visual vocalization on CMU Seasons Extended dataset [8]. Performances of network matcher pairs are compared. NN means Euclidean distance-based nearest neighbor matching. LGlue refers to LightGlue [86]. Nov. means the network was trained using rendered novel views of our NeRFs.

Scenes	CMU Seasons Extended [8]			
	Traj.	Urban	Suburban	Park
SuperPoint+LGlue	96.1 / 98.8 / 99.6	93.3 / 96.0 / 97.9	89.1 / 92.7 / 95.0	
R2D2+NN	94.4 / 97.3 / 99.0	85.3 / 88.9 / 93.5	74.2 / 78.8 / 83.6	
iCDC-R2D2+NN	96.5 / 98.8 / 99.4	93.4 / 94.9 / 97.1	87.5 / 90.3 / 92.9	
Nov.-iCDC-R2D2+NN	96.5 / 98.7 / 99.4	92.8 / 94.3 / 96.7	85.9 / 89.0 / 91.7	
Default-R2D2+(iCDC-LGlue)	86.4 / 90.5 / 94.9	77.7 / 83.5 / 92.3	53.2 / 60.5 / 74.5	
iCDC-R2D2+(iCDC-LGlue)	90.9 / 94.1 / 97.3	87.9 / 91.8 / 96.5	74.0 / 80.0 / 88.6	
Nov.-iCDC-R2D2+(iCDC-LGlue)	89.8 / 93.5 / 97.3	86.0 / 90.3 / 95.8	71.8 / 78.6 / 88.3	

Using the same setup, i.e., the same camera mounted on a car facing forward, these networks and matchers are nearly bridging the performance gap, even when applied to new scenes. This is partly due to the strength of the features, but also because they overfit to the matching patterns, limiting their ability to generalize to new setups. We demonstrate this through an ablation study presented in the long-term localization benchmarks in Tabs. 5.11, 5.12, and 5.13. SuperPoint+LightGlue outperforms all other network-matcher combinations, with only iCDC-R2D2+NN proving competitive across all datasets, except Aachen Day-Night. Without random cropping in the LightGlue training, performance significantly drops. Introducing cropping brings improvements, making the results comparable but not superior, except with RobotCar day-time data, which has characteristics similar to the 4Seasons setup.

Similar to our benchmark, training LightGlue with iCDC-R2D2 instead of default-

Table 5.12: Benchmarking long-term visual vocalization on RobotCar-Seasons dataset [90]. Performances of network matcher pairs are compared. NN means Euclidean distance-based nearest neighbor matching. LGlue refers to LightGlue [86]. Nov. means the network was trained using rendered novel views of our NeRFs.

Scenes	RobotCar-Seasons v2 [90]	
Traj.	Day All	Night All
SuperPoint+LGlue	63.9 / 93.8 / 99.3	24.5 / 45.9 / 56.6
R2D2+NN	64.5 / 94.4 / 99.2	16.8 / 32.6 / 39.4
iCDC-R2D2+NN	64.9 / 94.6 / 99.4	24.5 / 45.5 / 52.2
Nov.-iCDC-R2D2+NN	65.1 / 94.7 / 99.4	23.5 / 47.1 / 52.7
Default-R2D2+(iCDC-LGlue)	63.9 / 93.8 / 99.0	5.6 / 12.1 / 29.1
iCDC-R2D2+(iCDC-LGlue)	64.3 / 94.2 / 99.4	13.5 / 23.5 / 39.4
Nov.-iCDC-R2D2+(iCDC-LGlue)	64.4 / 94.6 / 99.6	16.8 / 31.9 / 49.2

Table 5.13: Benchmarking long-term visual vocalization on Aachen Day-Night [125]. Performances of network matcher pairs are compared. NN means Euclidean distance-based nearest neighbor matching. LGlue refers to LightGlue [86]. Nov. means the network was trained using rendered novel views of our NeRFs.

Scenes	Aachen Day-Night v1.1 [125]	
Traj.	Day	Night
SuperPoint+LGlue	90.0 / 96.0 / 99.3	75.9 / 90.6 / 98.4
R2D2+NN	88.0 / 95.3 / 97.9	67.0 / 86.4 / 96.3
iCDC-R2D2+NN	87.7 / 93.7 / 96.6	67.5 / 84.3 / 93.7
Nov.-iCDC-R2D2+NN	86.9 / 93.7 / 96.7	68.1 / 84.3 / 93.7
Default-R2D2+(iCDC-LGlue)	70.3 / 81.1 / 89.3	28.3 / 39.3 / 59.7
iCDC-R2D2+(iCDC-LGlue)	67.1 / 76.6 / 86.4	27.7 / 39.8 / 62.3
Nov.-iCDC-R2D2+(iCDC-LGlue)	69.8 / 80.6 / 88.7	37.2 / 50.8 / 71.7

R2D2 leads to meaningful performance improvements, except in Aachen Day-Night. Integrating novel views into training enhances performance in Aachen Day-Night as well as in RobotCar Night, but, surprisingly, it results in a performance decline in CMU.

Additional results on the long-term localization benchmark We detailed our key findings in Sec. 5.7.5, where we evaluated our extractors on the long-term localization benchmark. In Tabs. 5.14 and 5.15, we present more detailed results for the same networks on the CMU Seasons [8] and RobotCar-Seasons benchmarks [90] respectively. In the CMU-Seasons benchmark, iCDC-R2D2 significantly outperforms other networks across all visual domains. Conversely, the results are more balanced in the RobotCar-Seasons benchmark. Echoing the less detailed results in the main chapter, iCDC-R2D2 excels in the day-time scenarios, although not uniformly across all visual domains. In the main results, notable improvements were observed in the night-time scenarios. These detailed results show that the greatest performance improvement occurs in the rainy night-time scenario.

Table 5.14: Detailed CMU Seasons [8] benchmark results

	SuperPoint [34]	R2D2 [114]	iCDC-R2D2		SILK [50]	iCDC-SILK
Overcast	85.1 / 87.9 / 92.7	87.7 / 90.9 / 94.2	93.7 / 95.5 / 97.0		39.3 / 43.8 / 52.3	80.4 / 83.3 / 87.1
Sunny	74.5 / 79.0 / 87.0	77.2 / 82.1 / 87.5	87.6 / 91.4 / 94.7		33.6 / 37.8 / 44.0	72.0 / 76.5 / 81.2
Foliage	78.0 / 81.7 / 88.8	80.7 / 85.0 / 89.6	89.2 / 92.4 / 95.2		35.9 / 40.1 / 47.1	74.4 / 78.2 / 82.6
Mixed Foliage	84.7 / 87.7 / 92.4	85.2 / 89.0 / 92.9	94.0 / 95.7 / 97.0		36.3 / 40.7 / 49.4	80.1 / 83.3 / 87.4
No Foliage	95.0 / 96.4 / 97.8	94.8 / 96.6 / 97.9	98.1 / 98.9 / 99.1		56.1 / 60.3 / 67.8	91.3 / 93.3 / 95.1
Low Sun	86.9 / 89.4 / 93.3	86.8 / 90.2 / 93.6	94.7 / 96.2 / 97.3		42.5 / 46.6 / 54.8	83.0 / 85.8 / 89.4
Cloudy	90.3 / 92.2 / 95.9	91.6 / 94.1 / 96.2	95.6 / 96.8 / 98.1		44.8 / 49.4 / 57.8	84.4 / 86.9 / 89.6
Snowy	91.1 / 93.0 / 95.7	90.8 / 93.5 / 95.8	96.9 / 97.7 / 98.2		43.1 / 47.3 / 56.4	85.5 / 88.2 / 91.0

Table 5.15: Detailed RobotCar-Seasons [90] benchmark results

	SuperPoint [34]	R2D2 [114]	iCDC-R2D2		SILK [50]	iCDC-SILK
Night Rain	12.8/23.6/27.1	16.3/30.0/33.5	31.0/49.8/54.2		10.3/16.7/20.2	15.3/30.0/36.5
Overcast Winter	57.3/97.0/100	58.5/ 98.2/100	59.1/97.6/100		55.5/ 98.2/100	54.3/ 98.2/100
Sun	62.1/87.9/99.1	60.7/ 89.3/99.1	59.8/88.4/98.2		47.3/70.1/84.4	58.9/87.1/98.2
Rain	78.0/ 94.6/100	78.0/ 94.6/100	78.0/ 94.6/100		77.6/ 94.6/100	78.5/94.6/100
Snow	68.8/97.7/100	67.9/98.1/100	68.8/98.1/100		70.2/97.2/99.1	70.2/98.6/100
Dawn	56.8/91.6/96.5	57.3/91.6/96.5	60.4/ 95.2/100		60.8/93.4/98.7	61.2/95.2/100
Dusk	79.2/95.4/100	79.7/ 95.9/100	79.7/95.9/100		80.2/95.4/100	79.7/95.4/100
Night	15.0/31.4/39.8	17.3/35.0/44.7	18.6/41.6/50.4		14.6/19.9/23.0	15.5/23.9/32.3
Overcast Summer	47.9/92.9/98.6	50.2/94.3/99.5	48.8/93.4/98.1		44.5/89.6/92.4	48.3/90.0/96.7

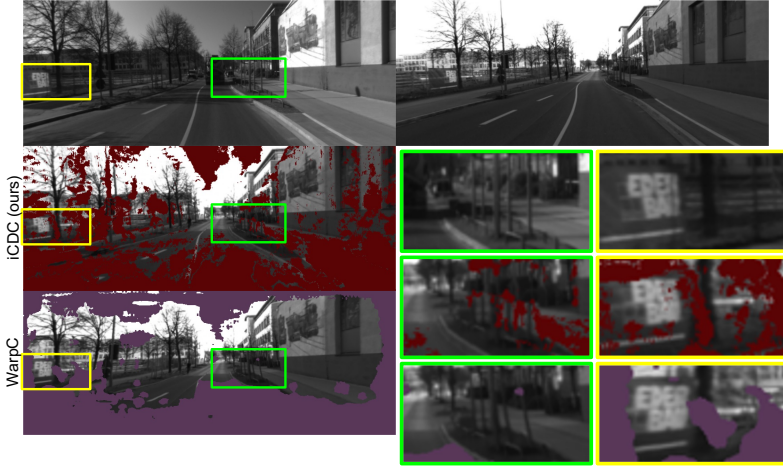


Figure 5.12: Additional comparison between our NeRF correspondences and WarpC [141] correspondences. Correspondences are generated for an image pair (top row). The left column’s second and third rows show WarpC and iCDC correspondences respectively. More specifically, the right-hand side image is warped onto the left-hand side image following the correspondences. We zoom in on windows for detailed examination. In the case of accurate correspondences, the warped frames align with the left-hand side frame.

Additional qualitative results. In Sec. 5.7.2, we presented results comparing correspondences generated by a dense correspondence network (WarpC [141]) and our

iCDC method. In Fig. 5.12 we present an additional example, highlighting the improved performance of our correspondence generation method across domains.

5.8.2 Additional Training Insights

NeRF training details. In Sec. 5.5, we outline our method of dividing each scene into individual blocks along each trajectory and training a smaller NeRF for each block. These smaller NeRFs are then combined per trajectory to form what we call a trajectory-NeRF. In our setup, however, we created a separate map for each trajectory pair to generate more refined relative poses across trajectories. To address scaling issues when comparing depth maps from different maps across trajectories, we train a trajectory-NeRF for each trajectory within every pair. Additionally, we extended each block with buffer zones to enhance the rendering quality of the NeRFs at block boundaries. This buffer region extends each block by 20% along each dimension.

In Instant-ngp [99], the representation space is divided into multiresolution hashed boxes, with each box divided into 128 discrete steps along each axis. During ray-casting, one point is sampled within each discrete step. All boxes share the same center but increase in size by multiples of two, starting from the smallest box. Therefore, the resolution of each discrete step in each box is a multiple of two smaller than in the smallest box. This multiscale encoding aims to accelerate training by sampling points at a higher rate near the center of the scene. However, to ensure thorough representation even in areas further from the center, the sampling point within each discrete step is randomized in each training iteration, allowing all regions to be adequately represented over multiple iterations.

For optimal results, it's necessary to maximize the number of relevant scene points within the smallest bounding boxes. For each block, we scale the scene so that the central bounding box (scale 1) spans 3 meters of the scene, along all axes. Consequently, the largest bounding box (scale 64) encompasses 192 meters of the scene, setting the limits for the NeRF representation. It's essential that the scene surfaces within each block are centered within these boxes. We aggregate 3D points from the map that correspond to each frame in a block, excluding any points that are more than 64 meters away from the related camera frames. We then center the scene around the median of the filtered point cloud. Additionally, to ensure a maximum number of points in the smallest boxes, we align the body diagonal of the boxes with the principal component of the point cloud.

This process is illustrated in Fig. 5.13. Furthermore, in Fig. 5.14, we present a collection of examples of the distributions of the point clouds centered around the bounding boxes. Excluding the points corresponding to the buffer regions, approximately 80% of the points are within the scale 16 bounding box.

Hyperparameters are selected per block to represent each block as accurately as possible. During training, some training runs encountered issues, getting stuck in local minima due to initial settings; this often led the NeRF to wrongly predict high density

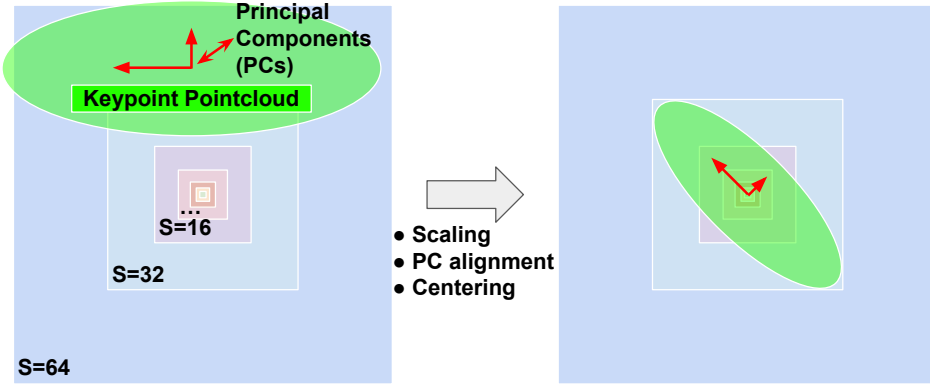


Figure 5.13: Optimized bounding box alignment. The keypoint point cloud is the set of 3D points a map corresponding to a scene block. Initially, the point clouds are scaled. Using Principal Component Analysis, we calculate the point cloud’s principal components. To fit as much of the scene into the smallest box possible, we center and transform the point cloud to align the principal components with the body diagonal of the boxes.

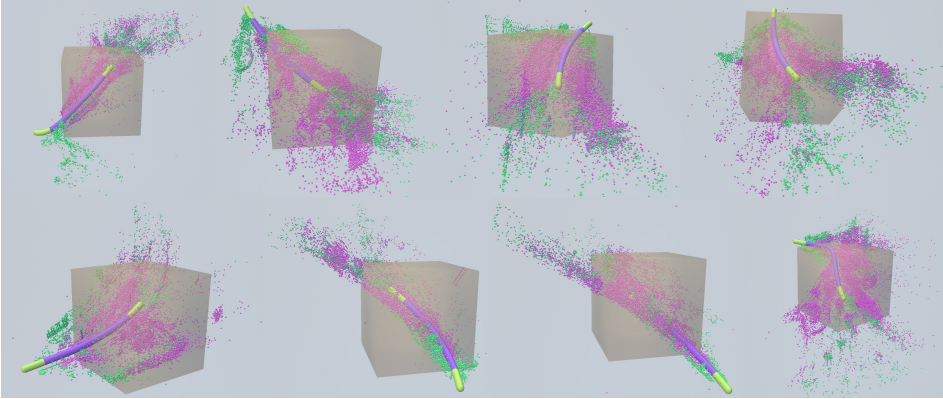


Figure 5.14: Bounding box alignment illustrations. Displayed are scenes aligned within scale 16 bounding boxes. Camera trajectories are depicted as tubes, with the point cloud representing the 3D scene for each camera frame. Frames within blocks are shown as pink point clouds and purple camera frames, while yellow camera frames and green point clouds depict those in the buffer region.

at the camera frame, resulting in zero depth estimations. To mitigate this, we reinitiate training several times during a single run, each time slightly adjusting the scene’s alignment within the bounding boxes. The best-trained NeRF is chosen based on its ability to accurately estimate depths that align closely with the ground truths at specific keypoints. These ground truths are the extracted depths from our maps, which were used for depth supervision.

If no training runs achieve an average depth error below a specified threshold, all the NeRFs from those runs are discarded. In such cases, training is either restarted with a new initialization or, after repeated failures, the blocks are abandoned; this latter outcome occurs in approximately 3% of blocks. We train each NeRF for $3 \cdot 10^5$ iterations.

5.9 Conclusions

Modern feature extraction methods push the boundaries of the field by developing advanced network architectures and innovative optimization techniques. These methods often involve supervising networks using perspective homographies and data augmentation to generate training tuples, allowing training schemes to operate flexibly in a self-supervised fashion. However, long-term localization scenarios introduce visual and perspective variations that exceed the capabilities of standard data augmentation techniques, creating a bottleneck that model-centric approaches cannot address. Thus, shifting towards enhanced data-centric approaches is crucial for successful life-long localization deployment.

We have established a new benchmark for long-term visual localization scenarios, facilitating the flexible assessment of visual localization performance across a wide range of visual domains. The 4Seasons dataset provides trajectories of images taken in diverse settings under various weather conditions, seasons, and lighting conditions during different times of the day. We produced highly accurate maps through joint trajectory SfM map refinement, providing ground truth relative poses between images from different visual domains. Our comprehensive evaluation of state-of-the-art extractors on this benchmark uncovered a substantial performance gap between intra- and cross-domain localization.

To bridge the gap towards accurate cross-domain localization, real cross-domain correspondences are needed to supervise networks. We employ available methods to generate correspondences across domains to accentuate this point. Using these correspondences, we trained two feature extractors, R2D2 and SiLK, significantly narrowing the performance gap. We introduce a novel data-centric approach, Implicit Cross-Domain Correspondences (iCDC), which generates accurate correspondences across visual domains by comparing the implicit 3D geometries of domain-specific NeRFs. By supervising networks with real cross-domain correspondences, we achieved a 36.1% reduction in the performance gap, as demonstrated in our relative pose estimation benchmark.

This research marks a significant advancement in enhancing the robustness and reliability of visual localization pipelines for lifelong deployment. It underscores the importance of data-centric methods in improving cross-domain localization performance. Future work could explore additional data-centric strategies and refine the efficiency and scalability of the proposed method. Foreexample, the recently introduced 3D Gaussian Splatting [71] could be deployed instead of NeRF to achieve faster fitting and run-time for scene representations. Developing robust cross-domain corre-

spondence generation pipelines that can handle dynamic objects within scenes would facilitate scaling to larger and more varied datasets. This progress would enable training networks capable of achieving intra-domain localization performance in long-term deployments. The insights from this study lay a solid foundation for further research aimed at optimizing the performance of feature extraction networks.

Chapter 6

Large-Scale Novel View Synthesis with Enhanced Neural Point-Based Graphics[†]

We present a new approach for novel view synthesis (NVS) in large-scale autonomous driving scenes. Our method is a neural point-based technique that leverages two modalities: images (cameras) and raw 3D point clouds (LiDAR). Current approaches encounter limitations in scalability and rendering quality, as directly using large point cloud representations for NVS in extensive scenes leads to degraded results. We identify the primary issue behind these low-quality renderings as a mismatch between geometry and appearance, stemming from using these two modalities together. To address this problem, we propose employing a connectivity graph between appearance and geometry, which retrieves points from a large point cloud observed from the current camera perspective and uses them for rendering. By leveraging this connectivity, our method significantly improves rendering quality and enhances run-time and scalability by using only a small subset of points from the large point cloud. Our approach associates neural descriptors with the points and uses them to synthesize views. To enhance the encoding of these descriptors and elevate rendering quality, we propose a joint adversarial and point rasterization training. During training, we pair an image-synthesizer network with a multi-resolution discriminator. At inference, we decouple them and use the image-synthesizer to generate novel views. Although our method is point-based, we also integrate our proposal into the recent 3D Gaussian Splatting work to highlight its benefits for improved rendering and scalability.

[†]This chapter is based on the work "Large-Scale Novel View Synthesis with Enhanced Neural Point-Based Graphics", submitted to IEEE Robotics and Automation Letters (RA-L) in July 2024 and will be available on arXiv. The article is adapted to fit the format and requirements of this dissertation.

6.1 Introduction

Rendering photo-realistic scenes has applications in many industries, including autonomous driving, gaming, cinematography, and virtual and augmented reality. Scaling novel view synthesis to large-scale scenes can drive advancements in these fields and potentially reduce costs, but it also presents significant challenges. This is particularly evident when using two different input modalities for rendering: images and point clouds. Images provide rich appearance information about the scene but lack geometric data. Conversely, LiDAR point clouds offer reliable geometric information. These two modalities can complement each other and enable several applications, especially novel view synthesis for large-scale scenes. In autonomous driving scenarios, it is common to use a combination of sensors, primarily cameras and LiDAR.

For fitting scenes and rendering new views, point-based graphics use points as the modeling primitive of the scene [79, 54]. Points and their associated attributes, such as surface orientation, disk radii, and color, are projected into the novel view to create rasterizations from which the view is rendered. Recent deep point-based graphics methods [2, 108] aim to encode local photometric and geometric parameters of the surface with neural descriptors, learned during optimization. However, these methods are typically applied to small objects with good coverage, often from 360-degree views.

In this paper, we propose a deep point-based graphics approach for novel view rendering in large-scale autonomous driving scenarios. Our method utilizes a map of the scene constructed from LiDAR scans and a set of reference images. Each point in the point cloud is associated with a descriptor, learned through a data-driven approach. For scene fitting and novel view prediction, the method retrieves the set of visible points from the point cloud to create rasterizations. These rasterizations are then mapped through a deep rendering network to produce RGB values, resulting in novel views.

In large-scale autonomous driving scenes, LiDAR scans are commonly accumulated to form a dense point cloud representation of the scene, which is very relevant in the scope of novel view synthesis. For example, projecting a single sparse LiDAR scan (synchronized with the camera) onto the camera canvas results in a low-quality rendering with gaps in the image due to the inherent sparsity of the rasterization. In contrast, accumulating point clouds from multiple LiDAR scans quickly and effectively addresses this sparsity issue and avoids the generalization limitations that a deep neural rendering network might encounter.

However, the projection of the scene point cloud into the image may result in an apparent conflict between what the image sees and what the projected point cloud (geometry) tells about the scene. In this paper, we point out that this results in a discrepancy between the view appearance and the view geometry. As a result, rendering

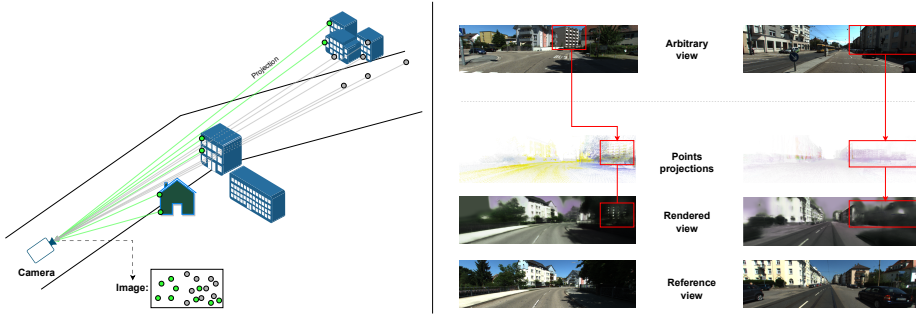


Figure 6.1: Left: to render the view from a certain camera perspective, 3D points of the scene are projected to the camera image. The resulting 2D projections correspond to 3D points that are seen by the camera (green points) as well as 3D points that are not seen from the current camera view (gray points). Using these projections to render a view results in low-quality images, as shown on the right side. Right: Examples of rendered views together with the actual reference view and 3D scene projections into the image. Both examples show incoherence between the points projections (geometry) and the reference view (appearance). As a result, rendering the view using these projections ends up in deteriorated views. The samples depict the projection of unseen buildings in the current camera view. That is, the obtained projections do not match the actual view as seen by the camera. As a consequence, the renderings are of low quality. In the first sample, incorrect projections propagated the unseen building in the reference view to the rendered view. In the other sample, this mismatch resulted in degraded rendering. The arbitrary views show the buildings which ended up in the projection of the camera.

novel views using the 3D point cloud directly, as in previous methods [2, 108], obtains low and degraded image quality. Fig. 6.1 illustrates the mismatching between the projected point cloud into the image and the actual view of the image. Points that are not observed in the current view are projected into the camera perspective, causing the rendered view to suffer from artifacts and fail to match the reference view.

To address the discrepancy between appearance and geometry, we propose establishing a connectivity relationship between the two input modalities. Specifically, we build a connectivity graph linking the RGB images from a camera with the 3D points from a LiDAR point cloud. Instead of projecting the entire point cloud into the image, we retrieve and rasterize only the 3D points relevant to the current view based on the connectivity information. This approach not only resolves the conflict between appearance and geometry, leading to improved renderings, but also accelerates the fitting and rendering process, making the pipeline more scalable. Additionally, this solution can be constructed once and used off-the-shelf for fitting as well as rendering new views in the given scene. For rendering a view from a novel pose, the 3D points from the closest pose in the connectivity graph are retrieved. This method does not rely on deep learning, thus avoiding the generalization limitations associated with learning algorithms.

To further improve the quality of renderings, we leverage Generative Adversarial

Networks (GANs) and propose a joint adversarial and point rasterization training. We propose to pair a multi-resolution discriminator with a U-Net image synthesizer during training. The image synthesizer processes rasterized neural descriptors to produce RGB images at different resolutions. Meanwhile, the discriminator directs the synthesizer to generate more realistic images and improve the source of neural descriptors. At inference, this pairing is decoupled, i.e., the discriminator is removed, and the synthesizer produces images.

In summary, our contributions are the following:

- We draw attention to the practical issue of incompatibility between appearance and geometry that emerges from large-scale rendering when using the two modalities as input: images and LiDAR point cloud.
- We propose to build a connectivity information graph between the two modalities, images, and LiDAR point cloud, to tackle the issue of conflict between appearance and geometry for the benefit of novel view fitting and rendering. The connectivity graph is built once and used off-the-shelf for fitting and rendering.
- We introduce a joint adversarial and point rasterization training method. This approach incorporates a multi-resolution image-based discriminator with a point-based image synthesizer network to improve neural encoding and rendering quality.

6.2 Related Work

Novel View Synthesis: Novel view synthesis has been a central research topic in the computer vision and graphics communities. Classic approaches leveraged the developments in structure-from-motion (SfM) [133, 127] and multi-view stereo (MVS) [51, 154] to enable novel view synthesis from a collection of images. These methods reproject and blend the input images into the novel camera perspective [40, 24, 58]. With the advent of deep learning, new techniques were introduced into novel view synthesis early on [44, 173, 73]. An influential stride in the field came with the introduction of Neural Radiance Fields (NeRFs) [96], where the entire scene is modeled continuously using fully connected layers. Since it is a ray-marching-based rendering method, the fully connected layers network is executed many times along the ray for each pixel of the view to be rendered. Following its introduction, many works were proposed to address NeRFs limitations such as fitting and rendering speed [49, 112, 100], quality [10, 170], and scalability [137, 143]. An alternative to the ray casting approach, forward rendering methods such as recent neural point-based graphics offer a strideable improvement in run-time [2, 108]. Point-based graphics methods provide a flexible and efficient way to work with 3D models, leveraging points as their fundamental building blocks. In essence, Point-based geometry representations allow the processing and visualization of 3D models without the need for costly surface reconstruction or triangulation. A very recent forward-rendering work, 3D Gaussian Splatting (3D GS), defines

a set of anisotropic Gaussians in a 3D world and performs adaptive density control to achieve reliable rendering results. However, these recent forward-rendering methods process and project the whole point cloud of a scene to synthesize a novel view. Our proposal is a forward-rendering approach that works directly on the raw point cloud. It associates low-dimension descriptors with each point to encode geometry and appearance. Rather than processing the whole point cloud of a scene, it retrieves only relevant scene points for novel view synthesis. It projects relevant points and the associated descriptors into the novel perspective, from which the novel view is rendered.

Point based graphics: Using point clouds to represent the scene for rendering instantiated interest early on due to its simplicity and efficiency. Point cloud forms an explicit scene representation and is directly indexable, making it suitable for forward rendering (for example, rasterization). In recent years, deep learning has been utilized to address point-based image rendering. The deferred neural rendering (DNR) [140] proposes to learn the point plenoptic function at different surface points, neural textures, jointly with a neural convolutional rendering network that maps the learned neural textures into an RGB image. Similar to DNR, NpbG [2] also learns neural descriptors to encode geometric and appearance features, together with a rendering network. However, NpbG uses raw point clouds to represent the geometry, avoiding the need for surface estimation and meshing. Rather than optimizing these neural descriptors, NpbG++ [108] proposes to estimate view-dependent descriptors by a convolutional feature extractor network in a way that reduces fitting time on new scenes. However, these methods work well for small objects and small-scale scenery. They suffer from artifacts and degradation when applied to large-scale scenes. This work proposes a neural renderer to tackle large-scale autonomous driving scenarios. The proposed method learns to optimize a set of neural descriptors that correspond to a large 3D raw point cloud.

Large scale scenes: Novel view synthesis for large scenes has been tackled by several methods relying on different scene representations. NeRF [96] models the scene in a multilayer perceptron function. However, this single function falls short in handling large-scale scenes. This limitation, among others like high fitting and run-time, pushed recent works [137, 143, 171] to learn multiple NeRFs by clustering the scene into different parts and assigning a NeRF representation for each partition. To know which NeRF representation to use, a retrieval criterion is learned as well. Recent neural point-based graphics (such as NpbG [2]) overcome these shortcomings, where novel views are synthesized directly with a forward pass of the convolutional network, making run-time much faster. Nevertheless, these works ([2, 108]) still act on small-scaled objects with enough view coverage. In the scope of tackling large-scale scenes, READ [83], a recent point-based graphics method, proposes a large-scale neural rendering method to synthesize autonomous driving scenarios. READ relies on a 3D point cloud that is obtained from SfM. However, reconstructing a large 3D scene using SfM is itself a time and memory-consuming process. Besides, it obtains sparse and noisy point clouds. This, in return, may require clustering of the scene to speed up scene reconstruction, fitting, and rendering time. READ suggests a learnable feature fusion module to fuse feature maps at different scales to tackle the sparsity of the 3D SfM point

cloud and the resulting gaps in the image projections. This may overfit to the order of the projected descriptors of the training images and limits its generalization capability. Rather than reverting to SfM, our method uses point clouds from LiDAR. LiDAR is a commonly used sensor in autonomous driving applications. To tackle sparsity, we accumulate LiDAR point clouds around (behind and ahead) the camera instance. This is done once as a preprocessing step, which densifies the point cloud and does not compromise the run-time of the method. Point-based graphics methods project the whole point cloud into the novel view and compute point visibility by employing the nearest rasterization scheme based on a Z-buffer, taking the closest points. In contrast, our method only projects 3D points that are relevant and observed in the current view. To achieve that, our method computes a connectivity relationship between the RGB images and 3D point clouds and retrieves the visible 3D points. Furthermore, we incorporate generative adversarial training into the point-based graphics modality.

GAN based rendering: GANs [52, 7, 91, 68, 161], usually consisting of a generator and a discriminator network, has been widely used for data generation as well as an implicit approach to model data distribution. Given the success of GANs in low-level vision tasks [155, 62, 74] and perception tasks [102, 25, 130, 129], researchers have recently explored to leverage the concept of GANs to boost image rendering. Sin-NeRF [162] utilizes a discriminator to improve the rendering quality. However, the discriminator only receives single-scale input patches, which lack the possibility to optimize the rendered image both locally and globally. MPR-GAN [163] downscale the rendered image into three resolutions and pass them to different discriminators. Nevertheless, the rendered image comes from a single scale generator and cannot optimize the rendered image progressively. On top of the NeRF head, GANeRF [115] introduces a generator that takes multiple downsampled versions of the same image and refines its quality as a final output. However, this extra generator will introduce computational complexity at both training and inference time. Instead, our design considers a multi-scale setting for both the rendering and the discriminator network, allowing progressive refinement at multiple levels but with only a single synthesizer network. This synthesizer network acts as the generator (renderer), with the discriminator being added only for scene fitting.

6.3 Method

Overview :

Our system is depicted in Fig. 6.2. Our system learns to produce images from novel perspectives for a given scene with a set of reference images, corresponding poses, intrinsic parameters, and associated LiDAR scans.

Given a point cloud of a scene with M points $G = \{\mathbf{g}_1, \dots, \mathbf{g}_M\}$, we associate to each point a C -dimensional neural descriptor (we set C to 8) $F = \{\mathbf{f}_1, \dots, \mathbf{f}_M\}$. The descriptors are learned in a data-driven approach to encode appearance and geometric information. For novel view rendering or scene fitting (learning these neural descriptors), our

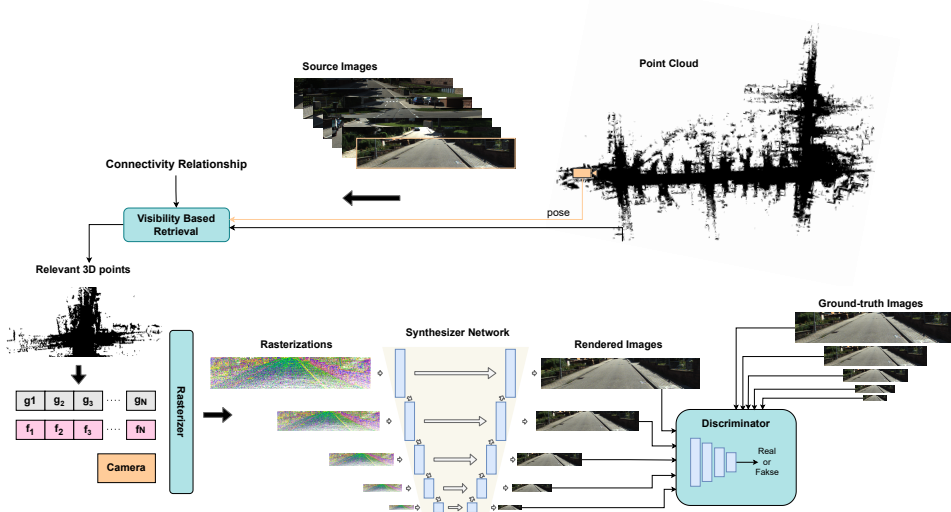


Figure 6.2: A sketched representation of our pipeline. A scene is represented by a set of source images and a point cloud. To render the view from a novel pose (yellow camera), a visibility retrieval module retrieves the 3D points that are relevant to the current view. The descriptors of the retrieved points are rasterized into the camera at different downsampled resolutions. Then, a refiner network renders the image by mapping the rasterizations into a novel view. At training time, a multi-resolution discriminator is deployed to improve rendering quality by classifying the generated images as real or fake. \mathbf{f}_i is the descriptor that corresponds to point \mathbf{g}_i

method first retrieves the visible points from the current camera perspective and then projects the 3D points together with their descriptors into images with different resolutions to form a pyramid of rasterized raw images. After that, the synthesizer, a deep network, maps the rasterized raw images into an RGB image. To update the weights of the synthesizer and optimize the neural descriptors, we apply an adversarial loss and a reconstruction loss that minimizes the difference between the predicted RGB images and the reference images. In the following, we describe each module of our system.

Visibility Estimation from Connectivity Relationship:

Some 3D points can not be seen from a camera’s perspective because of the structures’ occlusions and non-permeability. However, these points can still be projected onto the image plane because they fall within the camera’s field of view. This results in a mismatch between the visual appearance from that camera perspective and the geometry derived from projecting the point cloud. Consequently, the network struggles to learn how to generate visually appealing renderings (Fig. 6.1).

To tackle this issue, several solutions can be proposed, as illustrated in Figure 6.3. One solution is to filter out points with depth values that exceed a certain threshold

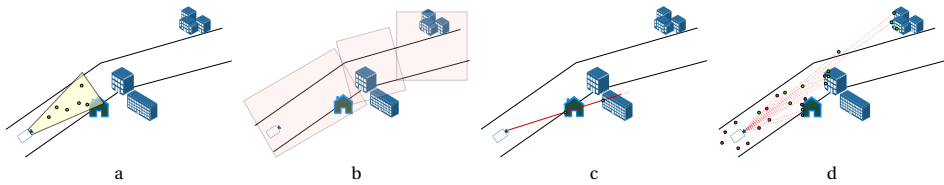


Figure 6.3: Different proposals to address large point cloud rendering: a) clipping points outside certain threshold, b) clustering of the scene, c) taking closest points (nearest z-buffer) d) Ours: building connectivity relationship between 3D points and poses/images. We compare the different proposals in section 6.3

from the perspective of a camera (Fig. 6.3 a). However, this is not practical in real and large-scale environments as this threshold has to be often adapted to the camera’s range of view. Besides that, in many scenarios, the image would see more than what the selected 3D points (points that fall within the clipped field of view) tell about the environment. The same issues arise by using a sliding window (rectangular or circular), which can be seen as an extension of this hard thresholding approach to include points behind the camera.

Instead of applying hard thresholds on camera perspective, a clustering of the scene (Fig. 6.3 b) can be applied/learned. The scene can be, for example, clustered based on the 3D points [143] or split into blocks [137]. However, this imposes additional complexities, such as dealing with overlapping boundaries and training different rendering networks for each scene division.

Another approach is to select the closest point, commonly used in previous neural point-based renderings [2, 108], as shown in Fig. 6.3 c. This method estimates a point’s visibility by creating a z-buffer and selecting the closest 3D point that projects to the same image pixel. While effective in small-scale scenes and helpful for handling occlusions, this solution requires dense pixel coverage from the 3D points visible in the current view, which is not feasible in large scenes.

Another solution is to learn to predict the visibility of a certain point from a certain camera perspective [137]. However, this imposes an additional deep network for visibility prediction. In addition to this, the learned visibility may not generalize well. In contrast to these solutions, we build a connectivity relationship between the RGB images and the 3D points from the point cloud. This relationship is not naturally available, as the two entities come from different sensor modalities.

In abstract terms, this connectivity relationship can be represented as a graph, where the vertices are the 3D points and the camera poses. An edge connects a 3D point to a pose if the point is visible from that camera perspective (image). We establish the connectivity relationship in two stages: a greedy stage followed by a pruning stage. First, we aggregate the LiDAR scans into a single point cloud for the scene, creating a compact representation with denser coverage from certain views. During the initial stage, connectivity edges are built incrementally as the 3D points are aggregated.

For an image I at timestamp t , we consider a number of LiDAR scans taken close to t , referred to as local isolation. This ensures scene overlap between the image and LiDAR at the current section of the scene, while excluding the rest of the scene from this stage of graph building. Consequently, all points in this locally isolated section are considered visible from the image. Specifically, we take n consecutive scans behind the camera and $2 \times n$ consecutive scans ahead of the camera. The scans behind the camera cover areas seen by the image (usually on the left, right, and ahead of the camera) that are not captured by the LiDAR scan closest to Camera I . Additionally, including previous scans densifies the point cloud, unlike using only the closest or upcoming scans.

After this graph building stage, the connectivity graph is considered generous as it associates all the 3D points that encapsulate the Camera to it. At fitting/rendering time, we follow a pruning stage. We apply pruning to account for local occlusions, in which case the 3D point lies within the Camera's field of view but is not actually seen from it.

For a given camera pose (during scene fitting or rendering), our method looks in the graph for the closest camera pose and retrieves all of its connected 3D points (during scene fitting, the queried pose exists in the graph). The pruning stage follows, in which only points that pass the visibility checks are kept, and the remaining points (i.e., edges) are dropped. The retrieved points are the visible points. These are the points that fall within the boundaries of the image and pass the depth checks. The depth checks include points that lie ahead of the camera (positive depth) and are the closest to the camera (shortest depth if multiple points fall onto the same pixel). For given $G^k = \{\mathbf{g}_i^k, \dots, \mathbf{g}_L^k\}$ and $D^k = \{d_i^k, \dots, d_L^k\}$ that denotes the set of 3D points that project to image k and the corresponding depths, respectively, these conditions can be expressed in the following equations:

$$h_{\text{positive}}(G^k, D^k) = \{\mathbf{g}_i^k \mid \mathbf{g}_i^k \in G^k \text{ and } d_i^k \in D^k \text{ and } d_i^k > 0\} \quad (6.1)$$

and

$$h_{\text{closest}}(G^k, D^k) = \{\mathbf{g}_i^k \mid \mathbf{g}_i^k \in G^k, d_i^k = \min\{d_j^k \mid \pi(\mathbf{g}_j^k) = (\mathbf{u}_i^k, \mathbf{v}_i^k) \text{ and } \mathbf{g}_j^k \in G^k\}\}, \quad (6.2)$$

where, functions $h_{\text{positive}}(G^k, D^k)$ and $h_{\text{closest}}(G^k, D^k)$ check if the projected 3D point \mathbf{g}_i^k has positive depth $d_i^k > 0$ and if d_i^k is the minimum depth among all points $\mathbf{g}_j^k \in G^k$ that project to the same image pixel $(\mathbf{u}_i^k, \mathbf{v}_i^k)$ with $d_j^k > 0$, respectively. $\pi(\mathbf{g})$ is the function that projects world coordinates \mathbf{g} into image pixel coordinates (\mathbf{u}, \mathbf{v}) . The two conditions can be combined in the following function:

$$h(G^k, D^k) = \{\mathbf{g}_i^k \mid \mathbf{g}_i^k \in G^k, d_i^k > 0, d_i^k = \min\{d_j^k \mid \pi(\mathbf{g}_j^k) = (\mathbf{u}_i^k, \mathbf{v}_i^k) \text{ and } \mathbf{g}_j^k \in G^k \text{ and } d_j^k > 0\}\}. \quad (6.3)$$

During scene fitting, only the descriptors of the retrieved points are updated in each iteration. To sum up, this connectivity graph is built once and can be used (off-the-shelf) for training and testing (rendering). The building stage applies local isolation, detaching the local scene from the whole scene and guaranteeing that far scene points that can project into the image (but not observed) are not visible by the current view. The pruning stage applies visibility checks to prune off occluded and locally unobserved points.

Rasterization:

After retrieving the visible point from the camera perspective to render, our system follows up to project the retrieved 3D points and the associated descriptors to a series of down-sampled resolutions to form a pyramid of rasterized raw images $\{\mathbf{S}\}_{t=1}^T$, where each raw image $\mathbf{S}_t \in \mathbb{R}^{\frac{H}{2^t} \times \frac{W}{2^t} \times C}$ (we set T to 5 in our experiments). The raw images compose the initial appearance information to be mapped, through the synthesizer, into the RGB values. The lowest resolution gets most filled by the projections and thus suffers from fewer holes and bleeding, while the highest resolution provides fine and detailed information. As the connectivity relationship graph retrieves the visible points, the corresponding visible descriptors are projected. This avoids updating descriptors of occluded points and unseen points during scene fitting.

Image Synthesis:

The synthesizer network, a U-Net architecture with gated convolutions, processes the rasterized raw images to produce the full-resolution RGB image with one forward pass. It takes the pyramid of the rasterizations as input and fuses the coarse and fine raw images at the respective resolution. The fusion process of fine and coarse details using the U-Net architecture aims to cover bleeding surfaces. Besides outputting the image at the original input (highest) resolution, we design the network to also obtain rendered views at the down-sampled resolutions of the input pyramid. This is aimed at coupling these renders with the multi-scale discriminator, which is explained next.

Joint Adversarial-rasterization training:

Apart from RGB regression, we also want the synthesizer network to learn generative capability so that it can still produce realistic outputs containing fewer artifacts even when clues from the point clouds are insufficient. Meanwhile, such generative capability helps the synthesizer generalize better on unseen inputs. Therefore, we utilize concepts from generative adversarial networks (GANs) to enhance the rendering quality of the point-based synthesizer. We propose a joint adversarial training in the point-based rasterization rendering pipeline. Specifically, we leverage an adversarial loss formulation in an end-to-end manner to impose additional constraints on the rendering pipeline. These additional constraints assist the training through gradient descent to push the rendered images to match the quality of the real ones, consequently updating the encoded geometric and appearance information in the neural descriptors.

We append the synthesizer network R_θ with a discriminator D_ϕ to match the generator-discriminator settings. Accordingly, the synthesizer network is deployed as the genera-

tor. Considering the fact that adding an extra generator for refinement would increase both training and inference complexity to our pipeline, we take advantage of adversarial training solely from the discriminator. In other words, the discriminator is only introduced for training but decoupled at test time. To further enhance the realism of our rendered output, we augment the discriminator with multi-scale branches, each operating on different image resolutions. In this setting, the discriminator is pushed during training to classify the input images as either real (the ground-truth RGB images) or fake (the generated images from the synthesizer network). Accordingly, the synthesizer is encouraged to fool the discriminator, thus improving its output quality. Adopting LSGAN [91] training criteria, our adversarial losses for the synthesizer and the discriminator can be formulated as

$$L_{adv}^{R_\theta} = \sum_{i=1}^{N_s} (D_\phi^{(i)}(R_\theta^{(i)}(\bigcup_{t=1}^T \mathbf{s}_t)) - 1)^2 \quad (6.4)$$

$$L_{adv}^{D_\phi} = \sum_{i=1}^{N_s} [D_\phi^{(i)}(R_\theta^{(i)}(\bigcup_{t=1}^T \mathbf{s}_t))^2 + (D_\phi^{(i)}(\mathbf{I}_{gt}^{(i)}) - 1)^2] \quad (6.5)$$

where i is the scale index and N_s denotes the total number of discriminator scales and $\mathbf{I}_{gt}^{(i)}$ is the scale-adjusted ground-truth image. In our experiments, we use N_s of 5.

Besides the adversarial loss, we use the perceptual loss [64] that computes the difference between the activations of a pretrained VGG network using the rendered and ground-truth images.

6.4 Experiments and Evaluation

6.4.1 Datasets

We use scenes from KITTI360 dataset for our experiments. It is a large dataset of real, rich driving scenarios. KITTI360 is released to push research on many interdisciplinary tasks, such as synthesizing novel view images. They address the shortcomings of the initial KITTI dataset by providing denser coverage of the scene and more accurate and geolocalized vehicle and camera poses. We select sequences from three different trajectories. For each sequence, we divide the scene into training and testing parts. We take every 10th frame as a testing frame. The rest constitutes the training set.

We build the point cloud of each scene by accumulating the 3D points from the LiDAR scans. This delivers a dense and compact 3D point cloud. In our experiments, we use a single GPU, a *Nvidia GeForce GTX TITAN X* with a capacity of 48 Gigabytes. Accordingly, we tune our selection of the sequence data size (images and 3D points)

to fit the given GPU space. For the KITTI360 dataset, we can select a sequence with slightly more than 300 frames and a number of 3D points close to 37 million.

Consequently, the derived dataset sequences are summarized in Tab. 6.1 and described below:

KITTI-0: we strip out a sequence of 316 camera frames (from frame 9602 to frame 9918) from dataset trajectory 0 with a total number of points close to 36.9 million.

KITTI-4: this corresponds to a part of trajectory 4 that starts with frame 9975 and ends with frame 10275, constituting a trajectory of 300 frames and 36.3 million points.

KITTI-6: we pick up a sequence of 310 frames starting at frame 8796 and ending at frame 9086 from dataset 6. This section constitutes a point cloud of around 37.4 million points.

Table 6.1: Details of the sequences from KITTI360 that are used in our experiments

	KITTI-0	KITTI-4	KITTI-6
Frames	9602 → 9918	9975 → 10275	8796 → 9106
Number of frames	316	300	310
Number of points	36950232	36352862	37376839

6.4.2 Baselines Methods

We compare our method to three state-of-the-art neural point-based rendering methods. These are:

- **Npbg:** a neural-point-based graphics approach aimed at per-scene optimization. It uses a raw point cloud as the geometric representation of the scene and augments each point with a learnable descriptor. We train their approach using the same input data as ours (LiDAR point cloud) [2].
- **Npbgpp:** a neural-point-based graphics approach. It leverages multiview observations and a point cloud of a static scene to predict view-dependent descriptors for the points through a neural network [108].
- **READ:** a neural-point-based graphics that is dedicated to large autonomous driving scenes. They propose a feature fusion module to fuse input data at different resolutions in the refining network as a way to deal with sparse point clouds [83].

6.4.3 Training Details

We conduct the experiments on the baselines using the same source of 3D data: accumulated LiDAR point clouds. We conduct our experiments using the full image as the

target (we do not crop the image into patches). All of the baselines use a U-Net-like architecture for regressing the novel view.

Table 6.2: Comparison of our method against the state-of-the-art neural point-based rendering methods on sequences from the KITTI360 dataset.

Method	KITTI-0			KITTI-4			KITTI-6		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
Npbg [2]	21.59	0.60	0.38	23.43	0.61	0.39	22.89	0.61	0.38
Npbgpp [108]	20.92	0.71	0.32	22.91	0.72	0.33	23.61	0.75	0.29
READ [83]	14.41	0.49	0.49	16.57	0.52	0.48	15.3	0.53	0.46
Ours	22.49	0.77	0.34	24.33	0.75	0.33	24.17	0.79	0.28

6.4.4 Comparison to Previous Neural Point Based Rendering Methods

Table 6.2 presents the quantitative results of our method and previous state-of-the-art methods using the standard rendering metrics: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM), and learned Perceptual Image Patch Similarity (LPIPS). As can be observed, our method reports better rendering metrics than other state-of-the-art methods. The main reason for that is our visibility solution, which addresses the incompatibility between appearance and geometry. It uses the visible points to render novel views. Previous methods estimate point visibility using different approaches, which show limitations in large-scale scenes. They compute a depth buffer projecting the point cloud into the image and taking the nearest points as the visible points. Npbg++ [108], in addition, rasterizes the point cloud into an image with reduced size, where the size is determined a priori by setting a visibility reduction factor to account for the sparsity of a point cloud. While taking the closest point can mitigate the impact of occluded parts, it is unreliable enough to address large point clouds and prevent the influence from unseen points. READ further implements a cube rasterization scheme to account for invalid regions in the image plane. Rather than projecting the 3D points themselves of a point cloud directly onto the image plane as in Npbg and Npbg++, it adapts the rasterization scheme by representing a 3D point with a cube. It divides the world space into regions with N voxels and projects the cubes instead of points into the image plane to reduce the impact of holes in the sparse point cloud and the resulting invalid regions in the image plane.

Rather than rasterizing the whole point cloud into the image, we rasterize the visible ones. Our connectivity graph shows its advantage in determining the visible points in the current frame and retrieving them without memory overhead. Our system, as a result, makes the geometry that is projected into the image and the appearance of the image compatible, which, in consequence, improves the rendering quality.

In Fig. 6.4, we complement these results with qualitative results by showing rendering results from novel views from our method and the recent state-of-the-art neural

point-based rendering methods. As can be observed, the renderings from our method are the ones closer to the reference images. Npbg++ [108], for example, produces white artifacts and unclear scenery. This is mainly due to the limitation of its visibility estimation approach.



Figure 6.4: Rendering results of our method compared to other neural point-based methods.

6.4.5 Usability and scalability of our method

In the previous experiment, we have shown the advances of our methods compared to other neural point-based rendering methods. Practically, we have selected 3D points and images to fit the maximum capacity of the GPU. In this experiment, we show the scalability of our method as well as its usability to other forward rendering approaches, specifically for the recent work, 3D Gaussians Splatting (3DGS) [71].

Introduction to 3DGS:

Splatting has been traditionally used for rendering by replacing each point with a 3D disk and projecting it onto an image. 3D Gaussians Splatting [71] has been recently proposed for explicit 3D scene representation. It has drawn much attention due to fast scene optimization and rendering time. 3DGS does not propose to train a deep network, as in recent deep learning-based methods, but optimizes a set of Gaussian parameters to represent the 3D scene. In fact, both 3DGS and neural point-based rendering are forward-rendering approaches that project the 3D representation to the image plane, i.e., rasterization.

A 3D scene can be described by a set of 3D Gaussians $\{G_i\}_{i=1}^M$, where the i -th Gaussian is defined as:

$$G_i = \{\mu_i, \mathbf{q}_i, \mathbf{s}_i, \mathbf{c}_i, \alpha_i\},$$

with $\mu_i \in \mathbb{R}^3$ being the center of the Gaussian, $\mathbf{s}_i \in \mathbb{R}^3$ and $\mathbf{q}_i \in SO(3)$ representing the scaling factor and the rotation quaternion, respectively, which define the covariance matrix $\Sigma \in \mathbb{R}^{3 \times 3}$. Additionally, $\mathbf{c}_i \in \mathbb{R}^3$ denotes the color, and $\alpha_i \in \mathbb{R}$ represents the opacity.

For a 3D query point $\mathbf{x} \in \mathbb{R}^3$, the Gaussian weight $g(\mathbf{x})$ is given by:

$$g(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)}, \quad (6.6)$$

where the symmetric 3D covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{3 \times 3}$ is defined as:

$$\boldsymbol{\Sigma} = \mathbf{R} \mathbf{S} \mathbf{S}^T \mathbf{R}^T. \quad (6.7)$$

Here, \mathbf{R} is a rotation matrix derived from \mathbf{q}_i , and \mathbf{S} is a diagonal matrix formed from the scaling factor s_i .

The process of 3DGS involves rasterizing these 3D Gaussians $\{G_i\}_{i=1}^M$ by sorting them in depth order in the camera space and projecting them onto the image plane.

When N Gaussians are projected onto a 2D point $\mathbf{p} \in \mathbb{R}^2$, the pixel color $C(\mathbf{p})$ is computed using α -blended rendering as follows:

$$C(\mathbf{p}) = \sum_{i \in N} c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j). \quad (6.8)$$

In the context of 3D scene reconstruction, the parameters of the 3D Gaussians $\{G_i\}_{i=1}^M$ are optimized by minimizing a reconstruction loss, which is derived from the rendering equation 6.8.

Our connectivity for 3DGS fitting:

We apply our connectivity-based visibility to 3D Gaussian Splatting [71] and evaluate scene fitting performance, i.e., using reference images. Similar to the previous experiment, we accumulate LiDAR points to build the point cloud of a scene. 3DGS [71] proposes to densify a point cloud with additional Gaussians to cover gaps in the reconstruction. Since the resulting point cloud is relatively dense, we execute 3DGS on the same single GPU without the densification option. In the second column of Tab. 6.3, we list the quantitative rendering metrics from scene fitting, with and without using our connectivity proposal, on a subset of KITTI-4, KITTI-4-reduced with a total number of 3D points of 19.4 million. We chose a subset of the scene to fit the available GPU memory when training 3DGS.

Table 6.3: Utilizing our connectivity relationship graph for 3D Gaussian Splatting [71] scene fitting.

Method	KITTI-4-reduced			KITTI-4		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
3DGS [71]	19.61	0.58	0.57	fail		
3DGS [71] + Our Connectivity	28.57	0.83	0.23	28.65	0.83	0.24

The table (column 2) shows a considerable improvement in the quality of scene fitting when using our connectivity setup. In the 3DGS setup (without our connectivity

relationship), a set of depth-ordered points overlapping the pixel, among them many unseen points, are blended to obtain the color. Whereas, by using our connectivity setup, only the retrieved visible points are blended to compute the color, which results in better optimization. We accompany this analysis with qualitative results in Fig. 6.5.



Figure 6.5: Using our proposed connectivity relationship drastically improves scene reconstruction using 3D Gaussians Splatting [71]

As can be observed, our visibility solutions enhance 3DGS scene fitting for large scenes.

Our connectivity for scalability:

3DGS optimizes not only the 3D points but also the associated Gaussian attributes, which require much additional memory. In the previous experiment, we performed the optimization using a subset of KITTI-4 to fit the available memory. In this experiment, we show the benefit of our connectivity relationship to scale the neural rendering to larger scenes.

Thus, we optimize 3DGS on the whole KITTI-4 section (36.3 million points), which is almost double the size of KITTI-4-reduced. Executing 3DGS on the KITTI-4 sequence ended in a failed reconstruction, as expected, due to the additional memory consumption needed for the optimization (last column of Tab. 6.3). However, the optimization was completed successfully by applying our connectivity relationship (3DGS + Our Connectivity). Our connectivity relationship retrieves only visible points, a small subset of the whole point cloud. This reduces computation and memory requirements, allowing for optimization of large scenes.

Our visibility for novel view synthesis:

In this section, we evaluate the quality of synthesized novel views from our neural point-based proposal and 3DGS [71]. Here, we use our connectivity setup for 3DGS [71]. Results are listed in Tab. 6.4.

We observe a significant difference in the quality of rendered novel views between our method and 3DGS, with our method outperforming 3DGS by a large margin. Additionally, compared to 3DGS's quality of scene reconstruction (as listed in Tab. 6.3), the quality of rendered views is inferior. We attribute this to the sparsity of views covering the scene in the KITTI360 dataset. 3DGS has demonstrated impressive quality in scenes where the camera typically obtains sufficient coverage by moving around an object. However, in scenes where the camera moves forward at car speed, the views

Table 6.4: Comparison of our neural point-based method and 3D Gaussian splatting (3DGS) for novel view synthesis on a sequence from the KITTI360 dataset. For 3DGS, we use our connectivity setup.

Method	KITTI-4 reduced		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
3DGS [71] + Our Connectivity	19.03	0.49	0.47
Ours	23.71	0.75	0.32

become sparse and provide less scene coverage. This is a research avenue that is being tackled as well.

In Fig. 6.6, we list samples of synthesized novel views from KITTI360 dataset. 3DGS obtains degraded quality.

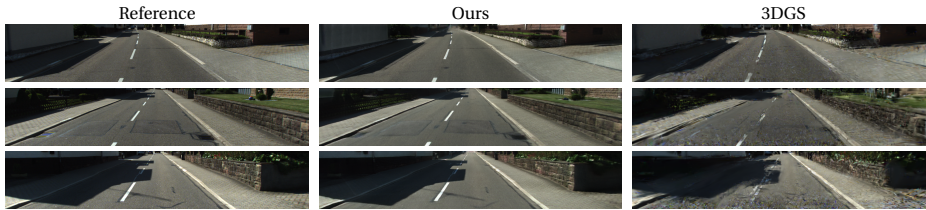


Figure 6.6: Our neural point based graphics methods obtains better renderings on Kitti360 than 3D Gaussian splatting [71].

Our connectivity for run-time:

We report further the run-time improvement from using our connectivity relationship. With the connectivity relationship, visible points are correctly estimated and used for fitting and rendering. This, as a result, accelerates the rendering function by more than 50 times as shown in Tab. 6.5

Table 6.5: Run-time improvements from using our connectivity graph on 3DGS.

Method	Points number	Render time (s) / FPS
3DGS	19403162	0.046 / 21.6
3DGS + Our Connectivity	720000	0.00089 / 1114

Using our connectivity drastically reduces the computation complexity by reducing the number of processed points.

6.5 Conclusions

We presented a novel neural point-based rendering solution for large-scale autonomous driving scenes that learns neural encodings of 3D points in a data-driven manner, utilizing a point cloud scene representation from LiDAR and reference views of the scene. These neural encodings are rasterized into the novel camera perspective and then mapped through a deep network into the RGB image. Our system, based on three key contributions, first identifies the primary cause of artifacts in neural point graphics when combining data from LiDAR point clouds and camera images, which arises from the incompatibility between the appearance of the reference image and the rasterized geometry. Second, we develop a solution to align these data sources by building a connectivity graph that associates images/poses with the 3D points seen from them, allowing our system to render only the visible points, thus accelerating run-time and reducing memory requirements, as validated by applying this to the 3D Gaussian Splatting. Third, we enhance the neural encodings and improve rendering quality using an adversarial generator-discriminator setup, pairing our generator function, which maps rasterized neural encodings into an RGB image, with a multiscale discriminator. In future, we would like to extend our approach by incorporating language embeddings and semantics into the neural encodings to facilitate text-conditioned novel view synthesis, scene understanding, and scene manipulation.

Chapter 7

Conclusions and Future Directions

This chapter summarizes the thesis work and contributions to the domains of computer vision and robotics, focusing on camera localization from a single image. We discuss the merits and limitations of the proposed solutions and close this chapter with proposals for new directions and future extensions of our work.

7.1 Conclusions

In this PhD dissertation, we have addressed the problem of camera localization. Specifically, we aimed to use deep learning to improve the accuracy of camera localization under the constraints of localizing from a single image while maintaining real runtime. While the focus is on improving localization, improvements in mapping are accompanied by improvements in localization. We have addressed the two directions of localization: direct and indirect.

In Chapter 2, we propose a solution for the indirect camera localization from a single image by proposing PixSelect. PixSelect is a deep learning method that works by assessing the importance of certain scene geometric information to localization and selecting reliable scene information. It computes the camera's pose from the scene's reliable geometric information. It considers that the scene is composed of discriminative and non-discriminative regions (ex, dynamic objects, sky, trees, bushes, repetitive and noninformative areas such as pavements, ...) and learns geometric scene priors accordingly. For learning geometric prior, PixSelect utilizes the reliable and geometrically verified scene reconstruction from SfM using images from different lighting conditions

and with different scene dynamics (dynamic objects). Our experiments show that by avoiding utilizing non-reliable regions, the outliers are minimized. By minimizing outliers, Pixselect improves localization accuracy and relies on fewer correspondences, resulting in run-time improvements.

In Chapter 3, we propose to create relative geometric information from the available labels and use them to improve the localization accuracy of indirect methods. The deep network acts as the mapping and localization engine in data-driven localization. In other words, the weights of the deep network encode the map of the environment and the localization pipeline. We thus propose to encode the geometric information of the scene into the deep network weights to improve localization from a single image. To integrate additional geometric constraints, we propose to use the available labels in the form of absolute poses to create relative poses that span the spatio-temporal space of the scene. Thus, we propose to compute relative geometric constraints not only from the cameras that are close in space and time (i.e., adjacent) but also from cameras that are distant in the space and time of a given scene. In our experiments, we show that incorporating additional geometric constraints improves the accuracy of camera localization from a single image.

As learned from Chapter 3, the incorporation of geometric constraints is necessary to improve localization for indirect methods. Thus, we propose in Chapter 4 to use the minimum available labels (i.e., absolute poses) to learn the geometric map of the scene in the form of a point cloud. That is, we propose to use absolute poses as the only supervision signal to obtain the 3D coordinates of the scene without explicit ground-truth labels of these coordinates. To illustrate, we exploit the minimum existing labels for the camera localization problem to guide a deep network to learn two representations of the map in two coordinate systems: the global reference frame of the scene and the camera's own coordinate system. For this purpose, we utilize a parameter-free and differentiable rigid alignment to pass gradients through the deep neural network to adjust its weights and continually learn these representations without explicit ground-truth labels. We accompany this learning setup with two additional losses: a reprojection loss to constrain the 3D global coordinates to the 2D image pixels and a consistency loss to make the two geometric representations consistent with the geometric pose. At inference, the deep network obtains the map of the scene in the two reference frames and obtains a pose by aligning them. In summary, our method uses the same labels as pose regression methods but computes a pose geometrically by aligning the two map estimates. As a result of this geometric computation, our method shows improvements in localization accuracy.

In Chapter 5, we improve localization by making it robust to long-term illumination changes such as daytime, seasonal, and weather changes. At first, we observed a limitation in the availability of evaluation pipelines in the research community to assess the robustness of existing feature extraction and description methods to these long-term changes. To cover this gap, we build a benchmark to enable flexible evaluation of visual localization performance across a broad spectrum of visual illumination changes. We build the proposed benchmark upon the existing 4Seasons by joint

refinement of trajectories from various day-time, weather, and seasonal conditions. Our proposed setup obtains highly accurate maps and provides ground-truth relative poses between images that are captured across different conditions. Following along, we use our benchmark to evaluate the performance of state-of-the-art feature extractors and descriptors under long-term illumination changes. This evaluation reveals a large performance gap between localizing under the same conditions and localizing across different illumination changes. To improve localization across long-term changes, we thus propose a data-driven approach to generate accurate and reliable correspondences across the different viewing conditions. To generate these correspondences, we propose representing the same environment with different implicit representations (based on neural radiance fields Nerfs), each corresponding to a different illumination condition. Consequently, we use these accurate correspondences to supervise feature extraction and description and reduce the performance gap by around 36%, as shown by our experiments. Our work sets a significant stride towards enabling long-term localization and enhancing its robustness to long-term visual changes.

In Chapter 6, we delve into a new direction to create additional source data to obtain more coverage of the scene and eventually improve localization. This work forms the initial step towards that goal. Manually driving around to generate a denser coverage of the scene is expensive. To address this, we aim to synthesize new views that are not existing in the dataset. To this end, we propose in Chapter 6 a novel neural point-based novel view synthesis method for large-scale autonomous driving scenarios. The proposed work learns neural encodings of the 3D points relying on a point cloud representation and reference views of the scene. From a novel perspective, our proposed work projects visible points and their corresponding neural descriptors into the new camera perspective and maps these into RGB values utilizing a refiner U-Net-like network. To reach the goal of generating reliable novel views, we first point out that the incompatibility between appearance and geometry is the root cause behind degraded synthesized views. To tackle this issue, we propose to create a connectivity graph that connects the reference perspectives to the corresponding observed 3D points. For learning the descriptors and synthesizing novel views, only the visible points are retrieved, that is, rasterized. By using the visible points instead of the whole point cloud, our system accelerates the runtime of the rendering function and reduces its memory requirement. For scene fitting, our proposed method merges adversarial training setup into point-based rendering to improve the quality of rendered views. In future work, we would like to improve upon this work to generate novel views to be used for data-driven localization.

7.2 Future Work

Camera localization is an ongoing research topic. The rise of deep learning has led to the use of data-driven approaches to solve camera localization and improve certain aspects of it.

In Chapter 2, we proposed PixSelect, which regresses the 3D scene and selects the reliable pixels that are useful for localization. The design was motivated by the research topic at hand: camera localization from a single image only. In fact, PixSelect can be regarded as keypoints detector. Nevertheless, to use it off-the-shelf for reconstructing the 3D scene (mapping) and localizing (example: SLAM), it is required to enhance PixSelect with description capability (i.e., obtain descriptors). A future line of work is to accompany the detection branch of PixSelect with a descriptor branch. This architecture is efficient since it allows for simultaneous detection and description as opposed to detect-then-describe classic approaches. Since our approach is data-driven, the training data is a vital factor in advancing this direction. In this regard, our method to create refined maps, which we developed in Chapter 5 as part of the evaluation benchmark, can be applied to other datasets and obtain reliable training data to train the detection and description of PixSelect. This will add robustness and generalization capabilities to PixSelect.

Acquiring data is essential for current data-driven approaches. In chapters 5 and 6, we have proposed novel methods to generate data, such as views from certain novel perspectives. Such generated data can be used, for example, to train and finetune a localization pipeline. As a future work, our proposed methods can take advantage of recent advancements in the field. For example, diffusion models can be used instead of GANs to enhance the quality of generated data. Diffusion models have shown better generative capability and quality than generative adversarial networks. Recently, 3D Gaussian splatting (3DGS) has shown impressive rendering quality with run-time much faster than NeRF. In fact, many works that were previously proposed based on NeRF are being enhanced with 3DGS. Similarly, we can utilize 3DGS as the basis for reconstructing the scene as a replacement for NeRF. This would enhance and accelerate our proposed methods.

While one of the goals behind our proposed methods in chapters 5 and 6 is data generation to tackle the topic addressed in this dissertation, there are other goals as well. One aspect is to embed language features in the created scene representations for scene understanding. Scene understanding enables many applications in robotics, virtual and augmented reality, and autonomous driving.

Developments in a certain field, such as Natural language processing, have inspired certain developments in computer vision and vice versa. Large language models LLMs such as LLaMA and Segment Anything Model (SAM) have set themselves as the foundation for many downstream tasks. In the context of our topic, local features have been used as the basis for data association to compute pose and other tasks. In the future, scene semantics such as segmentation and language could be used to associate data for pose estimation or change detection in maps. In this regard, an influential direction would utilize the rise of foundational models to inspire the creation of such models for the task of keypoints matching or pose estimation.

The developments in generative models are fast and influential. Advancements in text-to-image models are being applied to generate videos from text or images. Such a capability would inspire new directions in the field of mapping and localization. For

example, an image-to-video model can be used to update an old reconstructed model of a certain place, given a single image depicting the new changes.

List of Contributions

7.3 List of Publications

- **Mohammad Altillawi**, “PixSelect: Less but Reliable Pixels for Accurate and Efficient Localization”, 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 2022, pp. 4156-4162.
- **Mohammad Altillawi**, Zador Pataki, Shile Li and Ziyuan Liu, “Global Localization: Utilizing Relative Spatio-Temporal Geometric Constraints from Adjacent and Distant Cameras”, 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, MI, USA, 2023, pp. 3358-3365.
- **Mohammad Altillawi**, Shile Li, Sai Manoj Prakhya, Ziyuan Liu and Joan Serrat, “Implicit Learning of Scene Geometry From Poses for Global Localization”, in IEEE Robotics and Automation Letters, vol. 9, no. 2, pp. 955-962, Feb. 2024.
- Zador Pataki, **Mohammad Altillawi**, Menelaos Kanakis, Remi Pautrat, Fengyi Shen, Ziyuan Liu, Luc Van Gool and Marc Pollefeys, "Long-Term Invariant Local Features via Implicit Cross-Domain Correspondences", Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) in Nov. 2023. Available on arXiv preprint arXiv:2311.03345, Nov. 2023.
- **Mohammad Altillawi**, Sai Manoj Prakhya, Fengyi Shen, Shile Li, Jin Xin and Ziyuan Liu, “Large-Scale Novel View Synthesis with Enhanced Neural Point-Based Graphics”, Submitted to IEEE Robotics and Automation Letters (RA-L) in July 2024. It will be available on arXiv.

7.4 List of Patents

- **Mohammad Altillawi**, Ibrahim Hallfaoui, Onay Urfalioglu, “Data processing apparatus and method for determining a pose”. An application for a patent is submitted on 10.09.2021 with the number PCT/EP2021/074880. Patent is filed world-

wide and in the USA. Accessible at: <https://patentimages.storage.googleapis.com/76/02/cf/68dedad8eefc2e/W02023036431A1.pdf>

- **Mohammad Altillawi**, Shile Li, Ziyuan Liu, “Method for obtaining the global camera pose from a single image by aligning the implicitly predicted depth and 3D global coordinates which are learned from pose labels only”. An application for patent filing with application number PCT/EP2023/055749 is submitted on 07.03.2023 to the European patent Office.
- **Mohammad Altillawi**, Ziyuan Liu, “Method for obtaining camera pose by utilizing relative pose constraints from adjacent and distant cameras covering the spatio-temporal space of a scene”. An application for patent filing with application number PCT/EP2023/076871 is submitted on 28.09.2023 to the European patent Office.

Bibliography

- [1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from [tensorflow.org](https://www.tensorflow.org).
- [2] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 696–712, Cham, 2020. Springer International Publishing.
- [3] Mohammad Altillawi. PixSelect: Less but Reliable Pixels for Accurate and Efficient Localization. In *IEEE International Conference on Robotics and Automation*, pages 4156–4162, 2022.
- [4] Mohammad Altillawi, Zador Pataki, Shile Li, , and Ziyuan Liu. Global Localization: Utilizing Relative Spatio-Temporal Geometric Constraints from Adjacent and Distant Cameras. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023.
- [5] Asha Anoosheh, Torsten Sattler, Radu Timofte, Marc Pollefeys, and Luc Van Gool. Night-to-day image translation for retrieval-based localization. In *International Conference on Robotics and Automation (ICRA)*, 2019.
- [6] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2016.
- [7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

- [8] Hernán Badino, Daniel Huber, and Takeo Kanade. Visual topometric localization. In *IEEE Intelligent vehicles symposium (IV)*, 2011.
- [9] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017.
- [10] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021.
- [11] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Eur. Conf. Comput. Vis. (ECCV)*, 2006.
- [12] Hunter Blanton, Scott Workman, and Nathan Jacobs. A Structure-Aware Method for Direct Pose Estimation. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 205–214, 2022.
- [13] Eric Brachmann. 12scenes_rendered_depth.tar.gz. In *DSAC* Visual Re-Localization [Data]*. heiDATA, 2020.
- [14] Eric Brachmann. 7scenes_rendered_depth.tar.gz. In *DSAC* Visual Re-Localization [Data]*. heiDATA, 2020.
- [15] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC-Differentiable RANSAC for camera localization. In *CVPR*, 2017.
- [16] Eric Brachmann and Carsten Rother. Learning less is more - 6D camera localization via 3D surface regression. In *CVPR*, 2018.
- [17] Eric Brachmann and Carsten Rother. Neural- Guided RANSAC: Learning where to sample model hypotheses. In *Proc. IEEE Int. Conf. Comput. Vis.*, 2019.
- [18] Eric Brachmann and Carsten Rother. Visual Camera Re-Localization From RGB and RGB-D Images Using DSAC. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5847–5865, 2022.
- [19] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.
- [20] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-Aware Learning of Maps for Camera Localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2018.
- [21] David Brüggemann, Christos Sakaridis, Prune Truong, and Luc Van Gool. Refign: Align and refine for adaptation of semantic segmentation to adverse conditions. In *Winter Conference on Applications of Computer Vision (WACV)*, 2023.

- [22] Mingpeng Cai, Chunhua Shen, and Ian D. Reid. A Hybrid Probabilistic Model for Camera Relocalization. In *British Machine Vision Conference*, 2018.
- [23] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Eur. Conf. Comput. Vis. (ECCV)*, 2010.
- [24] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM transactions on graphics (TOG)*, 32(3):1–12, 2013.
- [25] Yun-Chun Chen, Yen-Yu Lin, Ming-Hsuan Yang, and Jia-Bin Huang. Crdoco: Pixel-level domain transfer with cross-domain consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1791–1800, 2019.
- [26] Zetao Chen, Adam Jacobson, Niko Sünderhauf, Ben Upcroft, Lingqiao Liu, Chunhua Shen, Ian Reid, and Michael Milford. Deep learning features at scale for visual place recognition. In *International Conference on Robotics and Automation (ICRA)*, 2017.
- [27] Zetao Chen, Lingqiao Liu, Inkyu Sa, Zongyuan Ge, and Margarita Chli. Learning context flexible attention model for long-term visual place recognition. *IEEE Robotics and Automation Letters (RA-L)*, 3, 2018.
- [28] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. In *NIPS*, 2016.
- [29] Ronald Clark, Sen Wang, Andrew Markham, Niki Trigoni, and Hongkai Wen. Vidloc: A Deep Spatio-Temporal Model for 6-DOF Video-Clip Relocalization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6856–6864, 2017.
- [30] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.
- [31] Gabriela Csurka. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint*, 2017.
- [32] Hal Daumé III. Frustratingly easy domain adaptation. *arXiv preprint*, 2009.
- [33] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2022.

- [34] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *IEEE Conf. Comput. Vis. Pattern Recog. Worksh. (CVPRW)*, 2018.
- [35] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.
- [36] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015.
- [37] Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. Adaptive recursive neural network for target-dependent Twitter sentiment classification. In Kristina Toutanova and Hua Wu, editors, *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 49–54, Baltimore, Maryland, June 2014. Association for Computational Linguistics.
- [38] Shi Dong, Ping Wang, and Khushnood Abbas. A survey on deep learning and its applications. *Computer Science Review*, 40:100379, 2021.
- [39] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.
- [40] Martin Eisemann, Bert De Decker, Marcus Magnor, Philippe Bekaert, Edilson De Aguiar, Naveed Ahmed, Christian Theobalt, and Anita Sellent. Floating textures. In *Computer graphics forum*, volume 27, pages 409–418. Wiley Online Library, 2008.
- [41] Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R Arabnia. A brief review of domain adaptation. *Advances in Data Science and Information Engineering: Proceedings from ICDATA 2020 and IKE 2020*.
- [42] Mohammed E Fathy, Quoc-Huy Tran, M Zeeshan Zia, Paul Vernaza, and Manmohan Chandraker. Hierarchical metric learning and matching for 2d and 3d geometric correspondences. In *Eur. Conf. Comput. Vis. (ECCV)*, 2018.
- [43] Martin A Fischler and Robert C Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, 1981.
- [44] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world’s imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524, 2016.

- [45] Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. *arXiv preprint*, 2017.
- [46] Gota Gando, Taiga Yamada, Haruhiko Sato, Satoshi Oyama, and Masahito Kurihara. Fine-tuning deep convolutional neural networks for distinguishing illustrations from photographs. *Expert Systems with Applications*, 66:295–301, 2016.
- [47] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(8):930–943, 2003.
- [48] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete Solution Classification for the Perspective-Three-Point Problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):930–943, 2003.
- [49] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14346–14355, 2021.
- [50] Pierre Gleize, Weiyao Wang, and Matt Feiszli. Silk—simple learned keypoints. *arXiv preprint*, 2023.
- [51] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M Seitz. Multi-view stereo for community photo collections. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.
- [52] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, 2014.
- [53] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. End-to-end learning of deep visual representations for image retrieval. *International Journal of Computer Vision*, 124(2):237–254, 2017.
- [54] J. P. Grossman and William J. Dally. Point sample rendering. In George Drettakis and Nelson Max, editors, *Rendering Techniques '98*, pages 181–192, Vienna, 1998. Springer Vienna.
- [55] Xiaoqing Guo, Chen Yang, Baopu Li, and Yixuan Yuan. Metacorrection: Domain-aware meta loss correction for unsupervised domain adaptation in semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2021.
- [56] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

- [57] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.
- [58] Peter Hedman, Tobias Ritschel, George Drettakis, and Gabriel Brostow. Scalable inside-out image-based rendering. *ACM Transactions on Graphics (TOG)*, 35(6):1–11, 2016.
- [59] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for Mobilenetv3. In *IEEE/CVF International Conference on Computer Vision*, 2019.
- [60] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint*, 2023.
- [61] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely Connected Convolutional Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.
- [62] Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, and Zhangyang Wang. Enlightengan: Deep light enhancement without paired supervision. *IEEE transactions on image processing*, 30:2340–2349, 2021.
- [63] Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image matching across wide baselines: From paper to practice. *Int. J. Comput. Vis. (IJCV)*, 129, 2021.
- [64] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part II 14*, pages 694–711. Springer, 2016.
- [65] Wolfgang Kabsch. A Solution for the Best Rotation to Relate Two Sets of Vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32:922–923, 1976.
- [66] Menelaos Kanakis, Simon Maurer, Matteo Spallanzani, Ajad Chhatkuli, and Luc Van Gool. Zippypoint: Fast interest point detection, description, and matching through mixed precision discretization. In *IEEE Conf. Comput. Vis. Pattern Recog. Worksh. (CVPRW)*, 2023.
- [67] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.

- [68] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.
- [69] Alex Kendall and Roberto Cipolla. Geometric Loss Functions for Camera Pose Regression with Deep Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5974–5983, 2017.
- [70] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *IEEE International Conference on Computer Vision*, pages 2938–2946, 2015.
- [71] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023.
- [72] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment Anything. *arXiv preprint arXiv:2304.02643*, 2023.
- [73] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, volume 40, pages 29–43. Wiley Online Library, 2021.
- [74] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8183–8192, 2018.
- [75] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 624–632, 2017.
- [76] Mans Larsson, Erik Stenborg, Lars Hammarstrand, Marc Pollefeys, Torsten Sattler, and Fredrik Kahl. A cross-season correspondence dataset for robust semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.
- [77] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An Accurate $O(n)$ Solution to the PnP Problem. *International Journal of Computer Vision*, 81:155–166, 2008.
- [78] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Int. Conf. Comput. Vis. (ICCV)*, 2011.
- [79] Marc Levoy and Turner Whitted. The use of points as a display primitive. 2000.
- [80] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018.

- [81] Xinyi Li and Haibin Ling. PoGO-Net: Pose Graph Optimization with Graph Neural Networks. In *IEEE/CVF International Conference on Computer Vision*, pages 5875–5885, 2021.
- [82] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018.
- [83] Zhuopeng Li, Lu Li, and Jianke Zhu. Read: Large-scale neural scene rendering for autonomous driving. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(2):1522–1529, Jun. 2023.
- [84] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Eur. Conf. Comput. Vis. (ECCV)*, 2014.
- [85] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. In *Int. Conf. Comput. Vis. (ICCV)*, 2021.
- [86] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. Lightglue: Local feature matching at light speed. *arXiv preprint*, 2023.
- [87] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.
- [88] David G Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis. (IJCV)*, 60, 2004.
- [89] Andrew L. Maas, Peng Qi, Ziang Xie, Awni Y. Hannun, Christopher T. Lengerich, Daniel Jurafsky, and Andrew Y. Ng. Building dnn acoustic models for large vocabulary speech recognition. *Computer Speech & Language*, 41:195–213, 2017.
- [90] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The oxford robotcar dataset. *The International Journal of Robotics Research*, 36, 2017.
- [91] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017.
- [92] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2021.

- [93] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 4040–4048, 2016.
- [94] Iaroslav Melekhov, Gabriel J Brostow, Juho Kannala, and Daniyar Turmukhambetov. Image stylization for robust features. *arXiv preprint*, 2020.
- [95] Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. Image-Based Localization Using Hourglass Networks. In *IEEE International Conference on Computer Vision Workshop*, pages 870–877, 2017.
- [96] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proc. European Conf. on Comput. Vis.*, pages 405–421, 2020.
- [97] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65, 2021.
- [98] Arthur Moreau, Nathan Piasco, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. CoordiNet: Uncertainty-Aware Pose Regressor for Reliable Vehicle Localization. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1848–1857, 2022.
- [99] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41, 2022.
- [100] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022.
- [101] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31, 2015.
- [102] Luigi Musto and Andrea Zinelli. Semantically adaptive image-to-image translation for domain adaptation of semantic segmentation. *arXiv preprint arXiv:2009.01166*, 2020.
- [103] Vinod Nair and Geoffrey E. Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Int. Conf. on Machine Learning, ICML’10*, page 807–814, Madison, WI, USA, 2010. Omnipress.
- [104] Tayyab Naseer and Wolfram Burgard. Deep Regression for Monocular Camera-Based 6-DOF Global Localization in Outdoor Environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1525–1530, 2017.

- [105] Felix Ott, Tobias Feigl, Christoffer Loffler, and Christopher Mutschler. ViPR: Visual-Odometry-Aided Pose Regression for 6DOF Camera Localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 42–43, 2020.
- [106] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
- [107] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models from Natural Language Supervision. In *International Conference on Machine Learning*, pages 8748–8763, 2021.
- [108] Ruslan Rakhimov, Andrei-Timotei Ardelean, Victor Lempitsky, and Evgeny Burnaev. Npbg++: Accelerating neural point-based graphics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15969–15979, June 2022.
- [109] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
- [110] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [111] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.
- [112] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14335–14345, 2021.
- [113] Jerome Revaud, Jon Almazán, Rafael S Rezende, and Cesar Roberto de Souza. Learning with average precision: Training image retrieval with a listwise loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5107–5116, 2019.
- [114] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2d2: Reliable and repeatable detector and descriptor. *Adv. Neural Inform. Process. Syst. (NeurIPS)*, 2019.

- [115] Barbara Roessle, Norman Müller, Lorenzo Porzi, Samuel Rota Bulò, Peter Kontschieder, and Matthias Nießner. Ganerf: Leveraging discriminators to optimize neural radiance fields. *arXiv preprint arXiv:2306.06044*, 2023.
- [116] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Int. Conf. Comput. Vis. (ICCV)*, 2011.
- [117] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019.
- [118] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2020.
- [119] Paul-Edouard Sarlin, Ajaykumar Unagar, Måns Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, and Torsten Sattler. Back to the Feature: Learning robust camera localization from pixels to pose. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 3246–3256, 2021.
- [120] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 3297–3307, 2019.
- [121] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *2011 International Conference on Computer Vision*, pages 667–674, 2011.
- [122] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Improving image-based localization by active correspondence search. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, pages 752–765, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
- [123] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9):1744–1756, 2017.
- [124] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, et al. Benchmarking 6dof outdoor visual localization in changing conditions. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018.
- [125] Torsten Sattler, Tobias Weyand, Bastian Leibe, and Leif Kobbelt. Image retrieval for image-based localization revisited. In *Brit. Mach. Vis. Conf. (BMVC)*, volume 1, 2012.
- [126] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2016.

- [127] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 4104–4113, 2016.
- [128] Yoli Shavit, Ron Ferens, and Yosi Keller. Learning Multi-Scene Absolute Pose Regression with Transformers. In *IEEE/CVF International Conference on Computer Vision*, pages 2713–2722, 2021.
- [129] Fengyi Shen, Akhil Gurram, Ziyuan Liu, He Wang, and Alois Knoll. Diga: Distil to generalize and then adapt for domain adaptive semantic segmentation. *arXiv preprint*, 2023.
- [130] Fengyi Shen, Akhil Gurram, Ahmet Faruk Tuna, Onay Urfalioglu, and Alois Knoll. Tridentadapt: Learning domain-invariance via source-target confrontation and self-induced cross-domain augmentation. *arXiv preprint arXiv:2111.15300*, 2021.
- [131] Fengyi Shen, Zador Pataki, Akhil Gurram, Ziyuan Liu, He Wang, and Alois Knoll. Loopda: Constructing self-loops to adapt nighttime semantic segmentation. In *Winter Conference on Applications of Computer Vision (WACV)*, 2023.
- [132] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, 2013.
- [133] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. 2006.
- [134] Noah Snavely, Steven M. Seitz, and Richard Szeliski. *Photo tourism: exploring photo collections in 3D*. Association for Computing Machinery, New York, NY, USA, 1 edition, 2023.
- [135] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2021.
- [136] Matthew Tancik, Vincent Casser, Xinchun Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2022.
- [137] Matthew Tancik, Vincent Casser, Xinchun Yan, Sabeek Pradhan, Ben P. Mildenhall, Pratul Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8238–8248, 2022.
- [138] Duyu Tang, Furu Wei, Bing Qin, Ting Liu, and Ming Zhou. Cooooll: A deep learning system for Twitter sentiment classification. In Preslav Nakov and Torsten

- Zesch, editors, *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 208–212, Dublin, Ireland, August 2014. Association for Computational Linguistics.
- [139] Jiexiong Tang, Hanme Kim, Vitor Guizilini, Sudeep Pillai, and Rares Ambrus. Neural outlier rejection for self-supervised keypoint learning. In *Int. Conf. Learn. Represent. (ICLR)*, 2020.
- [140] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *Acm Transactions on Graphics (TOG)*, 38(4):1–12, 2019.
- [141] Prune Truong, Martin Danelljan, Fisher Yu, and Luc Van Gool. Warp consistency for unsupervised learning of dense correspondences. In *Int. Conf. Comput. Vis. (ICCV)*, 2021.
- [142] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2022.
- [143] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12922–12931, 2022.
- [144] Michal Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. *Adv. Neural Inform. Process. Syst. (NeurIPS)*, 33, 2020.
- [145] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017.
- [146] Abhinav Valada, Noha Radwan, and Wolfram Burgard. Deep Auxiliary Learning for Visual Localization and Odometry. In *IEEE International Conference on Robotics and Automation*, pages 6939–6946, 2018.
- [147] Julien Valentin, Angela Dai, Matthias Nießner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin. Learning to Navigate the Energy Landscape. In *IEEE International Conference on 3D Vision*, pages 323–332, 2016.
- [148] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-Based Localization Using LSTMs for Structured Feature Correlation. In *IEEE International Conference on Computer Vision*, pages 627–637, 2017.
- [149] Bing Wang, Changhao Chen, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, and Andrew Markham. Atloc: Attention Guided Camera Localization. In *AAAI Conference on Artificial Intelligence*, volume 34, pages 10393–10401, 2020.

- [150] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clipnerf: Text-and-image driven manipulation of neural radiance fields. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2022.
- [151] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Minghui Tan, Xinggang Wang, et al. Deep High-Resolution Representation Learning for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(10):3349–3364, 2020.
- [152] Li Wang, Ting Liu, Gang Wang, Kap Luk Chan, and Qingxiong Yang. Video tracking using learned hierarchical features. *IEEE Transactions on Image Processing*, 24(4):1424–1435, 2015.
- [153] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring Large-Scale Vision Foundation Models with Deformable Convolutions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14408–14419, 2023.
- [154] Xiang Wang, Chen Wang, Bing Liu, Xiaoqing Zhou, Liang Zhang, Jin Zheng, and Xiao Bai. Multi-view stereo in the deep learning era: A comprehensive review. *Displays*, 70:102102, 2021.
- [155] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV) workshops*, pages 0–0, 2018.
- [156] Patrick Wenzel, Rui Wang, Nan Yang, Qing Cheng, Qadeer Khan, Lukas von Stumberg, Niclas Zeller, and Daniel Cremers. 4seasons: A cross-season dataset for multi-weather slam in autonomous driving. In *In Proceedings of the German Conference on Pattern Recognition*, 2020.
- [157] Tobias Weyand, Andre Araujo, Bingyi Cao, and Jack Sim. Google landmarks dataset v2-a large-scale benchmark for instance-level recognition and retrieval. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2020.
- [158] C. WU. Visualsfm: a visual structure from motion system,. <http://www.cs.washington.edu/homes/ccwu/vsfm>, 2011.
- [159] Jian Wu, Liwei Ma, and Xiaolin Hu. Delving Deeper into Convolutional Neural Networks for Camera Relocalization. In *IEEE International Conference on Robotics and Automation*, pages 5644–5651, 2017.
- [160] Xinyi Wu, Zhenyao Wu, Hao Guo, Lili Ju, and Song Wang. Dannet: A one-stage domain adaptation network for unsupervised nighttime semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2021.

- [161] Chulin Xie, Chuxin Wang, Bo Zhang, Hao Yang, Dong Chen, and Fang Wen. Style-based point generator with adversarial rendering for point cloud completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4619–4628, 2021.
- [162] Dejia Xu, Yifan Jiang, Peihao Wang, Zhiwen Fan, Humphrey Shi, and Zhangyang Wang. Sinnerf: Training neural radiance fields on complex scenes from a single image. In *European Conference on Computer Vision*, pages 736–753. Springer, 2022.
- [163] Qingyang Xu, Xuefeng Guan, Jun Cao, Yanli Ma, and Huayi Wu. Mpr-gan: A novel neural rendering framework for mls point cloud with deep generative learning. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–16, 2022.
- [164] Fei Xue, Xin Wang, Zike Yan, Qiuyuan Wang, Junqiu Wang, and Hongbin Zha. Local Supports Global: Deep Camera Relocalization with Sequence Enhancement. In *ICCV*, pages 2841–2850, 2019.
- [165] Fei Xue, Xin Wu, Shaojun Cai, and Junqiu Wang. Learning Multi-View Camera Relocalization With Graph Neural Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11372–11381, 2020.
- [166] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Tsung-Yi Lin, Alberto Rodriguez, and Phillip Isola. Nerf-supervision: Learning dense object descriptors from neural radiance fields. In *International Conference on Robotics and Automation (ICRA)*, 2022.
- [167] Yongbin You, Yanmin Qian, Tianxing He, and Kai Yu. An investigation on dnn-derived bottleneck features for gmm-hmm based robust speech recognition. In *2015 IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP)*, pages 30–34, 2015.
- [168] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.
- [169] Kaihua Zhang, Qingshan Liu, Yi Wu, and Ming-Hsuan Yang. Robust visual tracking via convolutional networks without training. *IEEE Transactions on Image Processing*, 25(4):1779–1792, 2016.
- [170] Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Nerfusion: Fusing radiance fields for large-scale scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5449–5458, 2022.
- [171] MI Zhenxing and Dan Xu. Switch-nerf: Learning scene decomposition with mixture of experts for large-scale neural radiance fields. In *The Eleventh International Conference on Learning Representations*, 2022.

- [172] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization in vision: A survey. *arXiv preprint*, 2021.
- [173] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 286–301. Springer, 2016.
- [174] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. *arXiv preprint*, 2023.
- [175] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2022.

