DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Artificial Intelligence and Computer Vision

Thesis title

Manuel Kolmet

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Artificial Intelligence and Computer Vision

Thesis title

Titel der Abschlussarbeit

Author:	Manuel Kolmet
Supervisor:	Prof. Dr. Laura Leal-Taixé
Advisor:	Qunjie Zhou
Submission Date:	November 1st, 2020

I confirm that this master's thesis in artificial intelligence and computer vision is my own work and I have documented all sources and material used.

Munich, November 1st, 2020 Manuel Kolmet

Acknowledgments

Abstract

This thesis introduces the concept of Visual-semantic Localization, which is to estimate an unknown query location using a database of known locations, given imagery and/or textual descriptions for each side. By combining the recent advances in Visual Localization and Multimodal Learning, we are, to the best of our knowledge, able to give the first evaluation of and framework for this sub-field of Artificial Intelligence.

The resulting contributions are threefold: First, we demonstrate that localization is possible across different modalities, e.g. estimating a location described as text based on a database of images, and introduce suitable models to bridge the modality gap. Secondly, we verify that the expected precision of that estimation can be improved by giving the query and/or database side in multiple, redundant modalities and develop models to unify this information. Thirdly, to approach the need for appropriately labeled data for our research, we have created three suitable datasets which are all made publicly available along with this document.

For our datasets, we extended the images in the ScanNet indoor dataset with descriptions in other modalities to form a real-world indoor dataset. Next, we rendered images from the Semantic3D point-cloud dataset extended by multi-modal descriptions in a similar fashion to form a real-world outdoor dataset. Lastly, we rendered virtual images from a photorealistic video-game with the needed descriptions to form a dataset that is artificial, but in it's nature similar to existing datasets for Visual Localization which are still pose challenges for even the most state-of-the art approaches.

Contents

Acknowledgments **iii**

Abstract **iv**

1 Introduction **1**
1.1 Section . . . 1
1.1.1 Subsection . . . 1

2 Related Work **3**
2.1 Visual Retrieval . . . 3
2.1.1 Classical approaches . . . 3
2.1.2 Learning-based approaches . . . 5
2.2 Visual Localization . . . 6
2.3 Cross-modal retrieval . . . 6
2.4 3D Visual Reconstruction . . . 8
2.5 Camera models and coordinate transformations . . . 9
2.6 Datasets . . . 10

3 Datasets for Visual-Semantic Localization **15**
3.1 Generating semantic descriptions for visual datasets . . . 15
3.1.1 Annotating real-world scenes using digital maps . . . 16
3.1.2 Generating semantic information from Object Detection . . . 16
3.1.3 Generating semantic descriptions from existing semantic labellings 16
3.2 Semantic ScanNet: Generating semantic descriptions for an indoor dataset 16
3.2.1 Subsection . . . 16
3.3 Semantic3D: Rendering and annotating images from labelled point-clouds 16
3.4 Los Santos Day & Night: Extracting Multi-modal data from a video-game 17

4 Implementation **18**
4.1 Visual Localization through Image Retrieval . . . 18
4.2 Semantic Localization . . . 18
4.2.1 Analytical Scene Graph localization . . . 18
4.2.2 Scene Graph localization using Geometric Learning . . . 18

4.2.3 Text-to-image localization 18
4.3 Combined Visual-Semantic Localization 18

5 Experiments **19**
5.1 Image Retrieval performance 19
5.2 The potential of Semantic Localization 19
5.3 Combining visual and semantic information 19

List of Figures **20**

List of Tables **21**

Bibliography **22**

1 Introduction

1.1 Section

Template citation:[Lam94]. My citation:[Jég+10]. Also, [Jég+10]

1.1.1 Subsection

See Table 1.1, Figure 1.1, Figure 1.2, Figure 1.3.

Table 1.1: An example for a simple table.

A	B	C	D
1	2	1	2
2	3	2	3

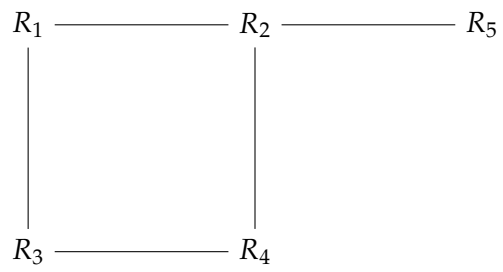

Figure 1.1: An example for a simple drawing.

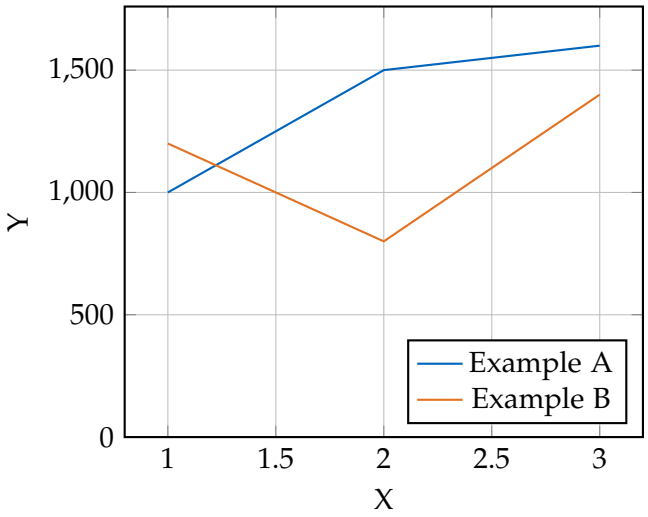

Figure 1.2: An example for a simple plot.

```
SELECT * FROM tbl WHERE tbl.str = "str"
```

Figure 1.3: An example for a source code listing.

2 Related Work

In this chapter, we will introduce the fundamental techniques in Computer Vision, retrieval, localization and cross-modal learning on which our own research is based. In the following sections, we will first introduce the basic methods for Visual Retrieval, which means to search a given database of images for those most similar to the input query image and return them ordered by a specified measure of similarity. Build upon these methods, we will then given an overview of the current field of Visual Localization, which means to estimate the real-world position of a query image, using no further information than a database of image-position pairs around the same location. As a basis for our own work, we will then introduce related work in the field of Cross-modal retrieval, which is to formulate a query in one modality (e.g. text) in order to receive the most similar matches of another modality (e.g. images.) Lastly, we give a brief overview of 3D Reconstruction and basic projective calculations used in our work, and present some of the available datasets for Computer Vision research.

2.1 Visual Retrieval

In order to search a database of images for those most similar to the query image, some measure of similarity is needed to reject those images that do not fit the query and to further rank the similarity of the positive matches. Ideally, this measure should be robust to basic variations such as changes in perspective, resolution and aspect ratio and possibly also to more advanced ones like time-of-day, occlusions, and changes in the structure of the scene itself. For some use-cases, the similarity measure also needs to capture essential elements in the images such as a type of scene or an object of interest and it might be necessary to extract this essential information in a compressed fashion to enable the search of large databases.

2.1.1 Classical approaches

Keypoint extraction Prior to Deep Learning-based approaches, Visual Retrieval was based on visual feature-extraction algorithms such as SIFT[Low99] and SURF[BTG06], that detect distinct image locations, so called *keypoints*. Each keypoint is accompanied by a corresponding fixed-sized list of numbers *("feature vector")* that describes the point

on an abstract level and is designed to be distinct for different keypoints but at the same time as invariant as possible for the same keypoint under visual variations. Based on these feature vectors, the extracted keypoints of different images can be compared in pairs, with a pair being declared as a *match* if the feature vector-difference is small enough. The number of matches can then be used to estimate the similarity between images and the knowledge of locations of (presumably) identical real-world points in multiple views can be further utilized for 3D reconstruction tasks.

Keypoint aggregation However, this approach would not be feasible when searching large data bases, given that the feature extraction itself, and especially the all-to-all keypoint matching procedure *("exhaustive matching")* between all images, implies a large computational effort. To remedy this problem, an aggregation procedure of these local keypoint descriptors into so-called vectors of locally aggregated descriptors or *"VLADs"* was introduced in [Jég+10]. The intention is to gather all the information about an image contained in a varying number of keypoints into a single, fixed-sized and compressed vector in such a way that the vectors, and therefore also the images themselves, can be compared and searched for efficiently. This procedure is done in four steps: As a preparation, a Codebook $C = \{c_1, ..., c_k\}, c_i \in \mathbb{R}^d$ of k visual words or *cluster centers* is learned using the K-Means algorithm. The information of a given image can then be accumulated by assigning each extracted feature vector $x \in \mathbb{R}^d$ to its closest visual word $c_i = NN(x)$. The VLAD descriptor $D \in \mathbb{R}^{k \times d}$ for this image is then created by summing up for each visual word its differences to the feature vectors assigned to it (using element-wise row subtractions):

$$D_i = \sum_{\{x | c_i = NN(x)\}} x - c_i \tag{2.1}$$

This descriptor matrix D is then transformed into the VLAD vector v by converting it into a vector and L_2-normalizing it. An efficient data base search is then possible using simple Euclidean distances between the vectors v, even for relatively small dimensions as for example with $d = 128$-dimensional SIFT feature vectors, and $k = 16$ learned visual words. Through an additional encoding procedure, the VLAD vectors can be further compressed to an order of 20 bytes and a correspondingly adapted search procedure allows the comparison between non-compressed vectors of a query image and compressed vectors of the database images, thereby enabling the desired feasibility for large data bases.

2.1.2 Learning-based approaches

As an equivalent to classical feature extraction and aggregation (e.g. SIFT features with VLAD aggregation), Visual Retrieval can also be performed purely via Deep Learning: In [Ara+15], the authors propose *NetVLAD*, a neural network layer that performs feature aggregation similar to VLAD, but is itself trainable together with an underlying convolutional network, which provides the actual image features to replace the keypoint extraction from the procedure above.

Re-iterating from above, the VLAD descriptor of an image, $D \in \mathbb{R}^{k \times d}$ is constructed by summing up for each learned visual word its differences to the feature vectors assigned to it. By writing out 2.1 explicitly for all columns and introducing the assignment function $a_k(x_i)$, which is 1 when cluster center c_k is the closest to x_i and 0 otherwise, the VLAD descriptor matrix is defined as $D(k,j) = \sum_i^N a_k(x_i)(x_i(j) - c_k(j))$. Similar to VLAD, the cluster centers are learned during training, but in order to also allow training of the base network, the equation has to be continuously differentiable. To this end, $a_k(x_i)$ is converted into a *Softmax* assignment, resulting in the NetVLAD descriptor

$$D(k,j) = \sum_{i=1}^{N} \frac{e^{\mathbf{w_k^T x_i} + b_k}}{\sum_{k'} e^{\mathbf{w_{k'}^T x_i} + b_{k'}}} (x_i(j) - c_k(j)) \tag{2.2}$$

with independent and trainable parameter sets $\mathbf{w_k}$, b_k and $\mathbf{c_k}$.

In order to train these parameters end-to-end together with the underlying feature extraction network for the visual localization task (formulated as image retrieval), the authors propose to use weakly annotated Google Street View Time Machine imagery. This data consists of panoramic images annotated with their GPS locations from different recording times. To obtain training data, one can pick a query image among these panoramas, and then retrieve potential positives from panoramas with a close-by geolocation but a different recording time, assuming that at least one of the potential positives shows the same location as the query image. A set of definite negatives can be generated from panoramas with distant geolocations (using the same or a different recording time as the query.) In order to train NetVLAD such that the resulting feature vector distance is small for images displaying the same scene and large for images displaying different scenes, the authors adapt the *Hinge loss* function to handle potential positives as $L_\theta = \sum_j max(0, min_i\, d_\theta^2(q, p_i^q) + m - d_\theta^2(q, n_j^q))$, using the training triplet $(q, \{p_i^q\}, \{n_j^q\})$, Euclidean distance $d_\theta(\cdot)$ and margin parameter m. While we train NetVLAD for image retrieval in a similar fashion, sets of definite positives are available in our case, thereby allowing us to use the common *triplet margin loss*.

In summary, the output of a NetVLAD network is again a fixed-sized vector containing the essential information of the image on an abstract level, designed so that the vectors of similar images are close to each other in Euclidean distance.

2.2 Visual Localization

The task of finding the position and orientation from where a query image was taken is named *Visual Localization*. In a classical, direct approach, a query image can be localized in a 3D Visual Reconstruction by extracting the querys keypoint descriptors and matching them to those of the reconstructed scene using an approximate nearest neighbor search, followed by a mechanism for outlier removal and pose estimation, e.g. RANSAC followed by a 3-Point solver [Li+12], which may lead to accurate camera pose estimates, but might also be unfeasible for larger datasets due to memory limits and ambiguities [Zho+19].

To overcome these challenges, the authors of [Zho+19] list further indirect approaches, which first use Visual Retrieval to find database images similar to the query. The query pose can then be estimated by matching its keypoints to only those keypoints of the full SfM model that are visible in the retrieved images, by matching to a small SfM model computed online from the retrieved images, or by directly estimating the relative pose between the query and the retrieved images through a 5-Point solver [Nis04]. [Zho+19] also show that currently, fully learning-based approaches to localization do not generally outperform these classical approaches (which might be partly based on machine learning themselves in the retrieval step.)

For our work, we pose Visual Localization as Image Retrieval, meaning that we retrieve the images that are most similar to the query, possibly perform an outlier removal step, and use the resulting average location as an approximation for the query location without further analysis.

2.3 Cross-modal retrieval

Similar to purely Visual Retrieval, retrieval can also be done across different *modalities* such as images, text descriptions, or Scene Graphs, meaning that the query is defined in one modality (e.g. text) and the database consists of items of another modality (e.g. images). Cross-modal retrieval then implies bridging the modality gap to then again retrieve those database items closest to the query in some measure of similarity, which can usually be done in both directions. For our work in Visual-semantic Localization, we also consider *multi-modal* retrieval, meaning that the query side is specified using two complementary modalities (e.g. an image and a Scene Graph) to retrieve the closest database items with a higher precision.

Unified Embeddings In order to define a fundamental formulation to bridge the modality gap between text and images, the authors of [KSZ14] present *Visual-Semantic*

Embeddings which project inputs from both modalities into a unified embedding space, such that inputs describing similar content produce close-by results in the embedding space, allowing for comparison and retrieval across the modality gap.

To project text captions into the embedding space, the words are first converted into vectors using a pre-computed word embedding matrix $W_T \in \mathbb{R}^{K \times V}$ with K the dimension of the embedding space and V the size of the vocabulary of known words, and a dictionary that specifies which word is converted into which column-vector of the matrix (including an entry for unknown words.) A complete sentence is then encoded by feeding the sequence of its word-vectors into an LSTM network [HS97] and using the last hidden state as the embedding representation.

The encoding of image inputs is done by first running them through a convolutional network trained for image classification and using one of its top layers as a feature vector. This feature vector is then converted into an embedding vector using a Fully Connected layer with weight matrix $W_I \in \mathbb{R}^{K \times D}$ with D the dimension of the feature vector.

The models LSTM parameters and those of the Fully Connected layer are then trained such that the image and text projections of tuples of images with their corresponding, correct description are projected to close-by vectors in the embedding space using a pairwise ranking loss. Using sets of pairwise-matching image and text embedding vectors $\mathbf{x}$ and $\mathbf{v}$, as well as "contrastive" sets of vectors $\mathbf{x}_k$ and $\mathbf{v}_k$ that are non-matching to all $\mathbf{x}$ and $\mathbf{v}$, the loss is defined as

$$\sum_{\mathbf{x}} \sum_{k} max(0, m - s(\mathbf{x}, \mathbf{v}) + s(\mathbf{x}, \mathbf{v}_k)) + \sum_{\mathbf{v}} \sum_{k} max(0, m - s(\mathbf{v}, \mathbf{x}) + s(\mathbf{v}, \mathbf{x}_k)) \tag{2.3}$$

with the scoring function $s(\mathbf{x}, \mathbf{v}) = \mathbf{x} \cdot \mathbf{v}$ (dot product) for normed vectors $\mathbf{x}$ and $\mathbf{v}$.

Scene Graphs In order to perform cross-modal image retrieval, the authors of [Joh+15] propose to formulate the query description through *Scene Graphs* instead of text descriptions to search a database for matching images. The goal is to find a way for complex and precise semantic descriptions based on a simple formulation, that can be evaluated without the need for advanced and possibly error-prone mechanisms for Natural Language Processing, which would be especially challenging in the presence of *co-refenrences*, e.g. if the same object-class occurs multiple times in a natural language description, it is hard to say whether this refers to the same, single object or if multiple objects of this class are present.

Formally, a Scene Graph can be defined as any basic graph structure by a set of Vertices V and a set of edges E. Given sets of class-labelled objects $O = \{o_1, ..., o_n\}$, relationship-types R and attribute-types A, the Scene Graph is then defined with objects, relationship-types and attribute-types as vertices $V = O \bigcup R \bigcup A$ and edges

assigning attributes to objects and connecting two objects through a type of relationship $E \subseteq (O \times A \bigcup O \times R \times O)$. In order to evaluate how well a given Scene Graph description matches an image, the authors also introduce the notion of a *Scene Graph grounding*: Using a set of candidate bounding boxes B found in the image, a possible grounding is then a map assigning each object in the Scene Graph to one of the bounding boxes $\gamma : O \to B$. As there are many possible ways to ground a Scene Graph to an image, an agreement measure is needed that first ranks all possible groundings of an image to find the best one, and then ranks between these best groundings of all the images to perform a top-k retrieval.

In the original paper, the authors formulate a *Conditional Random Field* and use the maximum a posteriori likelihood as an agreement measure. The term for a possible grounding γ is formulated using the objects O, edges E and candidate bounding-boxes B as $P(\gamma|O, E, B) = \prod_{o \in O} P(\gamma_o|o) \prod_{(o,r,o') \in E} P(\gamma_o, \gamma_{o'}|o, r, o')$ where γ_o is the bounding box assigned to o in γ. The unary potential measuring how well the assigned bounding box matches the objects class and attributes is then scored using R-CNN detectors trained on all occurring object classes and attribute types. The binary potential measuring how well the assigned bounding box pair matches the corresponding object classes and the relationship type r is scored using trained *Gaussian Mixture Models*.

To perform training and evaluation, the authors created a dataset of 5000 images manually annotated with corresponding Scene Graphs, but as this dataset is aimed at image retrieval rather than localization, we could not use it in our research. Nevertheless, we do use the idea of a Scene Graph as a fundamental, structured way to express and match semantic descriptions of complex scenes: During our automatic data annotation, we always first create Scene Graphs and then generate text descriptions from them. We also work directly with Scene Graphs through purely analytical Scene Graph-to-image scoring and through *Geometric Deep Learning*.

2.4 3D Visual Reconstruction

The 3-dimensional structure of a scene can be recovered based on a set of images that densely cover it using *Structure-from-Motion* ("SfM") software [SF16], which, broadly speaking, works by extracting keypoints from all the images and then incrementally finding keypoint matches between some of the images, estimating their relative poses, triangulating the keypoint positions in 3D and then adding more and more images to the current reconstruction. Most importantly for our work, the output of SfM consists of a point-cloud based on the triangulated 3D keypoints, a graph structure showing which of the keypoints are visible in which image and estimates for camera matrix and position for every image in the database in a scene-specific, relative coordinate system.

2.5 Camera models and coordinate transformations

In this section, we present basic coordinate system formulations and projections between them that are essential for our research.

Projections In Computer Vision algorithms, it is frequently necessary to project points between the 3D world and different viewpoints and image planes in it. For projective calculations, we will extend the 3D Cartesian coordinate system to the *Homogeneous coordinate system* by appending a 4th entry such that the Homogeneous point $\vec{p}_h = (X, Y, Z, W)$ represents the Cartesian point $\vec{p}_c = (X/W, Y/W, Z/W)$ with $W = 1$ in most cases.

A common simplification for visual projections is the *Pinhole Camera Model*, which assumes that a light ray from an object can only reach the cameras image plane, if it intersects with the 3D point of the camera center or its *focal point*. According to the Pinhole Model, the camera can be specified using an intrinsic matrix **I**, which captures characteristics of the camera itself, and an extrinsic matrix **E**, which captures information about cameras 3D position, specifically its rotation and translation. Matrix **I** can be specified using the cameras *focal lengths* f_x and f_y, its *optical center* coordinates c_x and c_y and a *skew factor* γ between its x- and y-axes. In simple cases, $c_x = w_I/2$ and $c_y = h_I/2$ with w_I,h_I the width and height of the image and $\gamma = 0$. Given either the focal lengths or the field-of-view angles of the camera, the other metric can then be derived using the equation $f_{x/y} = \frac{c_{x/y}}{tan(\alpha_{x/y}/2)}$ with α_x, α_y the total field-of-view angles (in degrees or radiants) of the camera, giving

$$\mathbf{I} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.4}$$

Matrix **E** specifies the rotation and translation for transforming a point from the world coordinate system into the camera coordinate system. Starting from $\mathbf{E} = [R|t]$, with $R \in \mathbb{R}^{3x3}$ the rotation matrix and $t \in R^3$ the translation vector, the matrix is then extend to preserve Homogeneous coordinates, giving

$$\mathbf{E} = \begin{bmatrix} r_{00} & r_{01} & r_{02} & t_x \\ r_{10} & r_{11} & r_{12} & t_y \\ r_{20} & r_{21} & r_{22} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.5}$$

Coordinate systems Figure 2.1 shows the three coordinate systems used in our work: *World Coordinates*, *Camera Coordinates* and *Image Coordinates*. The raw input data of our

datasets is given in 3D World Coordinates with respect to some origin and measured in a global unit (usually meters) with x and y as the floor plane (not necessarily aligned with the actual floor of a scene) and the z-axis pointing upwards. A point in World Coordinates can be transformed to Camera Coordinates using $\hat{p}_{cc} = E \cdot \hat{p}_{wc}$ with $\hat{p}$ indicating a point given in Homogeneous coordinates. The point is now specified with the camera center as origin, x- and y-axes aligned with the image plane and the z-axis pointing away from the camera along its *view vector*, all still measured in the global unit. It is noteworthy, that the x and y coordinates of $\hat{p}_{cc}$ equal the distance of point p from the camera center along these axes, but do not yet specify the location of *its projection onto the image plane*, meaning that two points p^1_{cc} and p^2_{cc} with different x and y coordinates can appear at the same location in the image. In order to have the x and y coordinates indicate the position of p's *projection*, we transform it again as $\tilde{p} = (\frac{\hat{p}_{cc,x}}{\hat{p}_{cc,z}}, \frac{\hat{p}_{cc,y}}{\hat{p}_{cc,z}}, \hat{p}_{cc,z})^T$, with $\tilde{p}$ indicating a point with its x- and y-coordinates projected onto the image plane. Lastly, a point can be further transformed into the Image Coordinate system, which then gives its x- and y-coordinates in pixels using $p_{ic} = I \cdot \tilde{p}_{cc}$, while the z-coordinate is unchanged and therefore still measured in the global unit, rather than in pixels. [1]

2.6 Datasets

Together with the recent advances in Visual Retrieval and Localization, numerous related datasets have been introduced for training and evaluation. In the following section, we present a selection of these datasets and illustrate the challenges in using them for our research, which mainly come down to missing semantic information.

Visual localization In [KGC15], the authors introduce the *Cambridge Landmarks* dataset consisting of over 12,000 images sampled from video recordings of six different outdoor scenes, automatically annotated with 6-DoF camera poses via Structure-from-Motion and split into testing and training sets. As this dataset is designed for Visual Localization, it does not include any semantic information. Automatically adding text annotations by running object detection networks (for example trained on *Cityscapes*) on these images failed, as the most prominent parts of the images are complex building architectures with very few other classes of static objects (such as lamps, traffic signs or vegetation), resulting in too few detections per image to generate descriptive annotations. Furthermore, automatically describing the architectures through *Facade parsing* [CSP14] appears unfeasible as multiple, un-segmented and un-rectified buildings with

[1] It might be necessary to flip the Image Coordinates depending on the directions of the different coordinate systems.

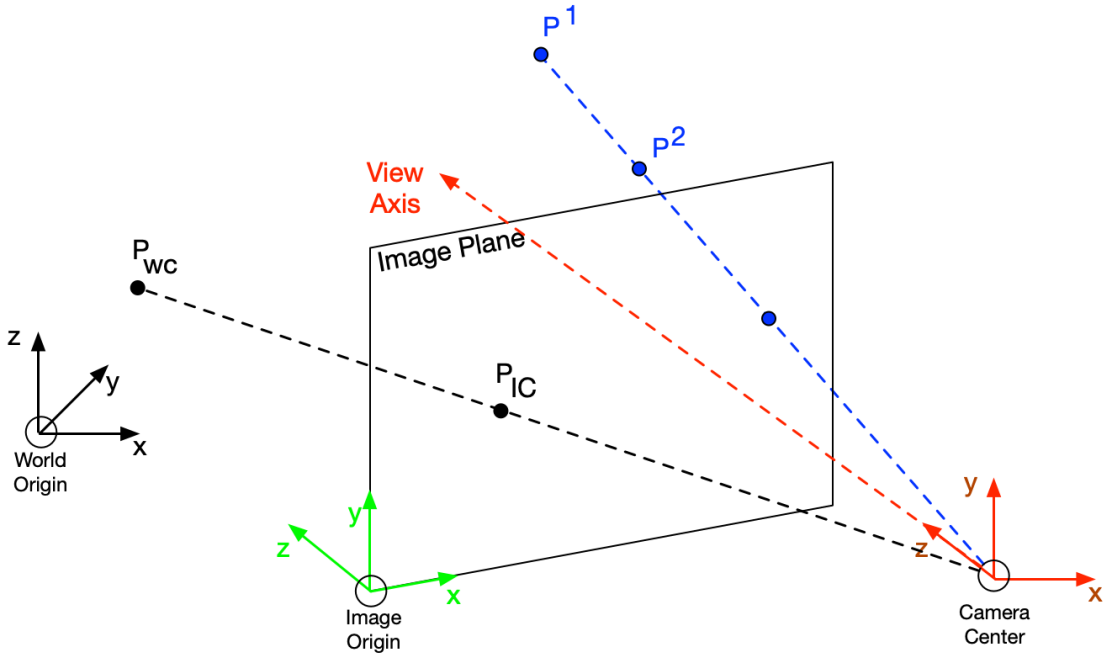

Figure 2.1: Visualization of the three coordinate systems used in our work. The blue dashed line and points give an example for two points with different world- and camera-coordinates, but an identical projection onto the image plane.

highly irregular structures occur in many of the images. Finally, our attempts to annotate the images using information about nearby places retrieved from online maps such as Google Maps also failed due to inaccuracies in the coordinate mappings, too few available nearby places, and due to the uncertainty about possible occlusions.

In addition to the Cambridge Landmarks scenes, the authors of [KGC15] also evaluate their localization system on the *7 Scenes* indoor dataset [Sho+13], which again is not annotated with semantic descriptions and no feasible way for automatic annotation presented itself.

Another, even more challenging dataset for Visual Localization is *Aachen Day & Night*, introduced in [Sat+12] consisting of 6,697 training and 1,015 query images with some of the query images taken at night time and even differences in resolution and aspect ratio between the images. While this dataset showed on average more object detections than Cambridge Landmarks, it was still not enough to generate precise descriptions for most of the scenes. Furthermore, descriptions solely based on detected objects did not match the most natural way for humans to describe these scenes, which would also be based on scene types ("big square", "small alley") and again descriptions of the visible facades

("glass wall", "medieval church"), with Facade parsing again seeming unfeasible. Lastly, the authors describe that they matched the Structure-from-Motion reconstruction of the images to an aerial view map of the streets retrieved from *Open Street Maps* [OSM] in order to provide their measurements in meters. While we were able to reproduce the mapping from image location coordinates to real-world GPS coordinates, annotating the images with descriptions from Open Street Map places was still unreliable, as the geo-location of a place can be ambivalent, e.g. whether it is specified as the center of its building, or as its doorstep, making it difficult to map the place precisely onto the image and to check whether it is occluded.

Lastly, the Tokyo 24/7 dataset from [Tor+15] consists of panoramic images downloaded from Google Maps [Inc] that can be used to localize freely captured query image in this city. In order to make the localization more robust and precise, the authors expand the database with synthesized views placed in between the original panoramas, thereby decreasing the expected distance between a given query image and the nearest database image. Again attempts to automatically annotate these images failed due to a lack of object detections and imprecise street map mappings, similar to other cases above.

Object Detection The research efforts in Deep Learning-based object detection and related tasks such as semantic segmentation, image captioning and cross-modal (e.g. text-to-image) retrieval gave rise to a variety of databases consisting of annotated images such as *Microsoft COCO* [Lin+14] and *PASCAL Visual Object Classes* [Eve+10]. While it would be straightforward to automatically generate semantic descriptions from these images if they are not already part of the dataset, it is clear that these datasets cannot be used for localization, as the images are captured from distinct and unconnected scenes.

Autonomous driving Due to recent interest in autonomous driving systems research, multiple datasets were published that contain vehicle-view images, commonly sampled from video recordings, together with location and even semantic annotations. The *Cityscapes* dataset [Cor+16] contains 5,000 images annotated with pixel-level semantic segmentations from 30 classes in addition to the GPS coordinates and vehicle orientation data and in *Semantic KITTI* [Beh+19], the authors provide similar vehicle-camera images together with point-cloud data that was reconstructed from a laser scanner next to the camera and point-wise labelled by hand using 28 classes. At a first glance, these datasets seem well suited for Visual-semantic Localization research, given that they consist of images from connected scenes annotated with location data and semantic labels that could be turned into semantic descriptions. On a closer look however,

multiple problems occur: Contrary to the most prominent example images from these datasets, many of the views are highway or road scenes, therefore including few non-moving object instances and many low-texture areas which make Visual Retrieval difficult. As our semantic descriptions should also only rely on static objects, the effective amount of available object classes is reduced considerably and the fixed setup of camera and scanner during capturing implies little variations in Azimuth and Elevation angles, while we would like to evaluate our system especially in the presence of viewpoint variations. This altogether makes autonomous driving datasets unsuitable for our case as well.

Point clouds In addition to the point-cloud reconstruction part of Semantic KITTI, we identified two point-cloud based datasets with semantic annotations that are suited for our research: *ScanNet* [Dai+17] contains 3D point-clouds of 707 distinct indoor scenes, reconstructed from 2.5 million views that were captured with RGB-D cameras. The point-clouds have been annotated with instance-level semantic segmentations and camera poses and calibration parameters are available for all views. This information made it possible to reliably re-project the 3D points and their semantic labels back into the original views, thereby allowing us to generate semantic descriptions for each view in a fully automatic fashion, making ScanNet one of the datasets in our evaluation.

In order to evaluate our system for Visual-semantic Localization on outdoor scenes, we also used the data from *Semantic3D.Net* [Hac+17], which consists of point-clouds for 30 outdoor scenes, each captured using a stationary terrestrial laser scanner. Of these 30 scenes, 15 were point-wise hand labelled using 8 classes of stationary objects[2]. Due to the stationary capturing procedure, the point-clouds exhibit empty spots where no objects were in range, scanning artifacts (e.g. when people walked in front of the scanner during capturing) and additional empty spots due to occlusions when the viewpoint is shifted away from the original scanner position. Nevertheless we were able to render images along various trajectories for each of the annotated scenes and re-project the labelled points back into them, again allowing us to generate semantic descriptions automatically and making Semantic3D.Net our main dataset for evaluation.

Artificial data As the combined requirements of a large dataset size (e.g. over 40,000 scans in SemanticKITTI) together with precise semantic annotations can imply an unfeasible amount of manual work, the idea of using artificially rendered datasets comes to mind. In the *SYNTHIA* dataset [Ros+16], the authors render images from an artificial city, with camera perspectives and semantic labels similar to Cityscapes using Unity [Tec]. The authors validate the usefulness of artificial data by including it in the

[2]Not counting the "unlabelled" class and considering cars as stationary

training process of semantic segmentation networks that are eventually evaluated on real-world data, which suggests that the usefulness of Visual-Semantic Localization networks can be confirmed or rejected on artificial data as well. Given that the camera parameters and trajectories can be specified freely during the Unity rendering process, it would be possible to specify them according to our needs to create an artificial database perfectly suited for Visual-semantic Localization. However, the authors of SYNTHIA did not release their underlying rendering pipeline, but only published the results as a new database of annotated images similar to Cityscapes.

Therefore, we also considered the approaches of [Ric+16] and [Krä18] where the authors modify the rendering processes of commercial video games to capture information additional to the games display output, which subsequently allows them to annotate the captured images with semantic and instance segmentations, depth maps, optical flow and camera positions fully automatically.

3 Datasets for Visual-Semantic Localization

The training and evaluation of our models and subsequent verification of our hypotheses is based on suitable data for model development and comparison. In the first section, we describe our main data requirements and a general strategy for automatic annotation. In the subsequent sections, we then explain our annotation processes in detail and present the three resulting datasets.

3.1 Generating semantic descriptions for visual datasets

To evaluate our hypothesis that localization performance can be improved by combining visual and semantic information, we require a dataset that contains images together with their respective semantic descriptions, which can be given in text form or via Scene Graphs. The dataset should be large enough to train Deep Neural Networks and be appropriate for visual localization, meaning that it should consist of a single, densely covered scene, contain enough textural information for image retrieval to work, but still pose enough challenges for state-of-the art systems to leave room for improvements.

Since we could not find a dataset that fulfilled all these requirements and also contained semantic descriptions for its images, our plan was to generate these descriptions for any of the existing Visual Localization datasets based on our learnings from section 2.6. We also decided that the annotation had to be done automatically, given that manual annotations would have required considerable human working hours which would have had to be redone if subsequent learnings had changed our data requirements.

This brought us to our general strategy for description generation, which is to automatically find information about the objects that are visible in a given image, most importantly their class, position and size, in order to then put them in localized relations with each other and finally turn these relations into text descriptions using a template-based language model.

3.1.1 Annotating real-world scenes using digital maps

3.1.2 Generating semantic information from Object Detection

3.1.3 Generating semantic descriptions from existing semantic labellings

3.2 Semantic ScanNet: Generating semantic descriptions for an indoor dataset

ScanNet

3.2.1 Subsection

sub 3.1

3.3 Semantic3D: Rendering and annotating images from labelled point-clouds

Sem3d

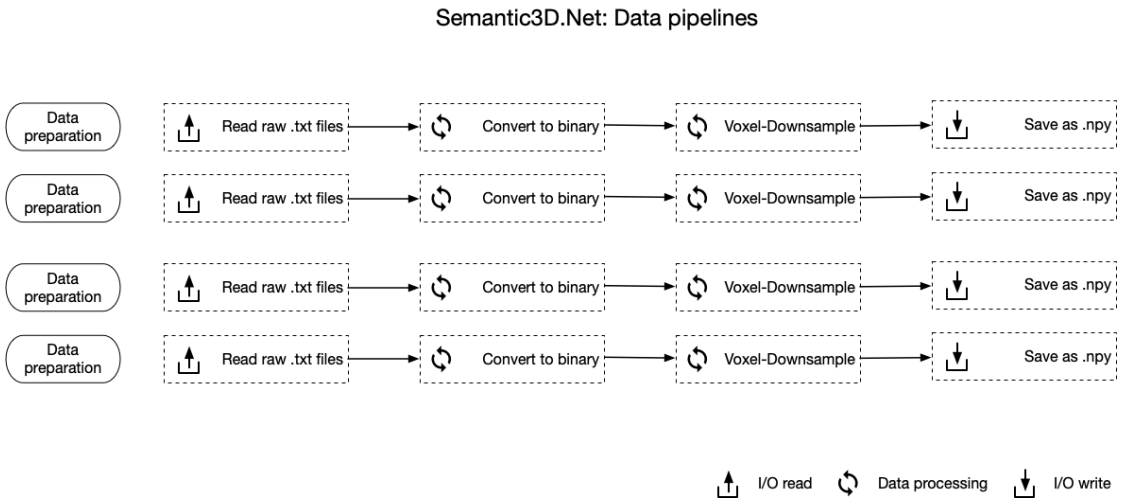

Figure 3.1: Examples for Visual Retrieval Examples for Visual Retrieval Examples for Visual Retrieval Examples for Visual Retrieval Examples for Visual Retrieval Examples for Visual Retrieval

3.4 Los Santos Day & Night: Extracting Multi-modal data from a video-game

GTA5

In the first section, we present the available datasets from related fields and demonstrate the challenges in using them for our research, which mainly come down to missing semantic information.

4 Implementation

4.1 Visual Localization through Image Retrieval

4.2 Semantic Localization

4.2.1 Analytical Scene Graph localization

4.2.2 Scene Graph localization using Geometric Learning

4.2.3 Text-to-image localization

4.3 Combined Visual-Semantic Localization

5 Experiments

5.1 Image Retrieval performance

5.2 The potential of Semantic Localization

5.3 Combining visual and semantic information

List of Figures

1.1 Example drawing 1
1.2 Example plot 2
1.3 Example listing 2

2.1 Visualization of the three coordinate systems used in our work. The blue dashed line and points give an example for two points with different world- and camera-coordinates, but an identical projection onto the image plane. 11

3.1 Examples for Visual Retrieval Examples for Visual Retrieval Examples for Visual Retrieval Examples for Visual Retrieval Examples for Visual Retrieval Examples for Visual Retrieval 16

List of Tables

1.1 Example table 1

Bibliography

[Ara+15] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. *NetVLAD: CNN architecture for weakly supervised place recognition*. 2015. arXiv: 1511.07247 [cs.CV].

[Beh+19] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences." In: *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*. 2019.

[BTG06] H. Bay, T. Tuytelaars, and L. V. Gool. "SURF: Speeded Up Robust Features." In: (2006).

[Cor+16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. "The Cityscapes Dataset for Semantic Urban Scene Understanding." In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[CSP14] A. Cohen, A. G. Schwing, and M. Pollefeys. "Efficient structured parsing of facades using dynamic programming." In: (2014).

[Dai+17] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. "ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes." In: *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*. 2017.

[Eve+10] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman. "The Pascal Visual Object Classes (VOC) Challenge." In: *Int. J. Comput. Vision* 88.2 (June 2010), pp. 303–338. ISSN: 0920-5691. DOI: 10.1007/s11263-009-0275-4.

[Hac+17] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys. "SEMANTIC3D.NET: A new large-scale point cloud classification benchmark." In: (2017).

[HS97] S. Hochreiter and J. Schmidhuber. "Long short-term memory." In: (1997).

[Inc] A. Inc. *Google Maps*. URL: www.google.com/maps.

[Jég+10] H. Jégou, M. Douze, C. Schmid, and P. Pérez. "Aggregating local descriptors into a compact image representation." In: (2010).

[Joh+15] J. Johnson, R. Krishna, M. Stark, L.-J. Li, D. A. Shamma, M. S. Bernstein, and L. Fei-Fei. "Image Retrieval using Scene Graphs." In: (2015).

[KGC15] A. Kendall, M. Grimes, and R. Cipolla. "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization." In: (2015).

[Krä18] P. Krähenbühl. "Free supervision from video games." In: (2018).

[KSZ14] R. Kiros, R. Salakhutdinov, and R. S. Zemel. *Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models*. 2014. arXiv: 1411.2539 [cs.LG].

[Lam94] L. Lamport. *LaTeX : A Documentation Preparation System User's Guide and Reference Manual*. Addison-Wesley Professional, 1994.

[Li+12] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. "Worldwide Pose Estimation using 3D Point Clouds." In: *European Conf. on Computer Vision*. 2012.

[Lin+14] T.-Y. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. "Microsoft COCO: Common Objects in Context." In: *CoRR* abs/1405.0312 (2014). arXiv: 1405.0312.

[Low99] D. G. Lowe. "Object Recognition from Local Scale-Invariant Features." In: (1999).

[Nis04] D. Nister. "An efficient solution to the five-point relative pose problem." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.6 (2004), pp. 756–770.

[OSM] O. F. (OSMF). *Open Street Map*. URL: www.openstreetmap.org.

[Ric+16] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. *Playing for Data: Ground Truth from Computer Games*. 2016. arXiv: 1608.02192 [cs.CV].

[Ros+16] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes." In: (2016).

[Sat+12] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. "Image Retrieval for Image-Based Localization Revisited." In: (2012).

[SF16] J. L. Schönberger and J.-M. Frahm. "Structure-from-Motion Revisited." In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[Sho+13] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. "Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images." In: *Proc. Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2013.

[Tec] U. Technologies. “Unity Development Platform.” In: ().

[Tor+15] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla. “24/7 place recognition by view synthesis.” In: (2015).

[Zho+19] Q. Zhou, T. Sattler, M. Pollefeys, and L. Leal-Taixe. *To Learn or Not to Learn: Visual Localization from Essential Matrices*. 2019. arXiv: `1908.01293 [cs.CV]`.