UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Semantic segmentation for visual indoor localization

Author:
Yurii KAMINSKYI

Supervisor:
Jiri SEDLAR, Ph.D.

*A thesis submitted in fulfillment of the requirements
for the degree of Master of Science*

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2019

Declaration of Authorship

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UKRAINIAN CATHOLIC UNIVERSITY

Abstract

Faculty of Applied Sciences

Master of Science

Semantic segmentation for visual indoor localization

by Yurii KAMINSKYI

The problem of visual localization and navigation in the 3D environment is a key to solving a vast variety of practical tasks. For example in robotics, where the machine is required to locate itself on the 3D map and steer to a specific location. Another example is a personal assistant in the form of a mobile phone or smart glasses that uses augmented reality techniques to navigate the user seamlessly in large indoor spaces such as airports, hospitals, shopping malls or office buildings.

The purpose of this work was to improve the performance of the InLoc localization pipeline that gives state-of-the-art results for indoor visual localization problem. That was done by developing relevant semantic features. Namely, we introduce a variety of features as a result of two different segmentation models: Mask R-CNN and CSAIL. We evaluate the quality of generated features and add the features of the better performing model into the InLoc localization pipeline.

With the introduced features we improved the performance of the InLoc localization pipeline and introduced approaches for further research.

Acknowledgements

I would like to thank Jiri Sedlar who supervised and directed my research for this thesis. Special thanks to Josef Civic(CIIRC CTU and INRIA Paris) and Hajime Taira(Tokyo Institute of Technology).

This work would not be possible without computational resources provided by CIIRC CTU.

I would like to thank UCU for having a chance to be the student at the master program of Data Science. I would also like to thank Intellias company for covering my tuition fees.

Contents

List of Figures

To my family and girlfriend

Chapter 1

Introduction

The purpose of this work is to improve the performance of the InLoc localization pipeline [Taira et al., 2018a], dedicated to the estimation of the exact camera pose on a large-scale 3D map. The goal of this thesis is to survey and evaluate different methods for indoor semantic segmentation with the aim of improving visual localization in indoor environments.

The problem of visual localization in space has an extensive application in different spheres: from the robotic systems trying to navigate through buildings to a specific location to an autonomous vehicle of any type that can park itself on a free slot and then drive back to a predetermined location.

In the previous years, the navigation in an outdoor environment is a very active field of research. There is a reason for that: the primary application for this field of research, autonomous driving, has a lot of working solutions which means that the research in this area is very vivid. In particular, there are many datasets of significant size and good quality available by [Cordts et al., 2016], and there are approaches and methods that can give a human-like performance. The fact that these solutions work in a real-world environment is a reliable indicator of the active research in this field.

The situation of the indoor localization is different as it is not so well developed. There are three main reasons for that. First of all, the main reason is the data. There are some datasets that contains indoor room layout, either collected visually by [Zhou et al., 2017] or generated by GANs [Song et al., 2017; Chang et al., 2017; Wu et al., 2018; Zhang et al., 2016]. However, these datasets are mainly devoted to representing the internal layout of one room or possibly a small apartment. But there are few datasets that contain consistent data of a whole building (e.g., an office space or a university campus). We will return to this topic in Chapter 2. The second reason why indoor localization is less explored is the complexity of the task itself. Complexity stands for the following.

One of the main reasons for the complexity of the indoor localization is the fact that even a small change of the viewpoint could significantly change the appearance [Taira et al., 2018b]. For example, if a camera that is capturing the corridor of the long building, is rotated by 30 degrees it will capture an entirely different scene, say printer near the corridor wall.

Secondly, the most obvious idea of the indoor localization is to have a reference objects in the image, which can easily give an understanding on the exact location of the camera, for example, the position of a window in the room or a painting on the wall. However, the problem is that there are "easily movable objects" in the environment which make a distraction for the algorithm that tries to find a correspondence in the particular images. The example of an object is people, chairs, small coffee tables. The issues described above will be addressed in this thesis.

The third issue is that indoor scenes often contain repetitive elements both on a large (same corridors and rooms) and small (same tables, doors) scale [Taira et al., 2018b]. The algorithm needs to have some other significant objects to refer.

Another critical issue is that the appearance of an indoor scene could change significantly over time, for example, changing of the lightning during the day and more permanent changes like changing the position of the chairs and desks in a classroom.

This thesis contributes to the performance of the existing InLoc approach by introducing additional features for a better understanding of the indoor environment. The features are introduced by applying different semantic segmentation algorithms. This work provides a comparison of the performance of different features that were introduced.

To sum up, this work is dedicated to helping solve the problem of indoor environment localization described above, with the constraints of a particular dataset and algorithmic pipeline, using semantic features.

Chapter 2

Datasets for semantic segmentation and indoor localization

In the introduction, we mentioned the problem of the right dataset for the task of indoor localization. Now we would like to give a quick overview of the datasets that were used in this work, and explain our choice of the datasets.

In this work, datasets were used for two different purposes: semantic segmentation and indoor localization. For the semantic segmentation, ADE20K and MS COCO were used, and for the indoor localization, we have used InLoc dataset. Below we will describe each of the mentioned datasets in detail.

2.1 Semantic segmentation datasets

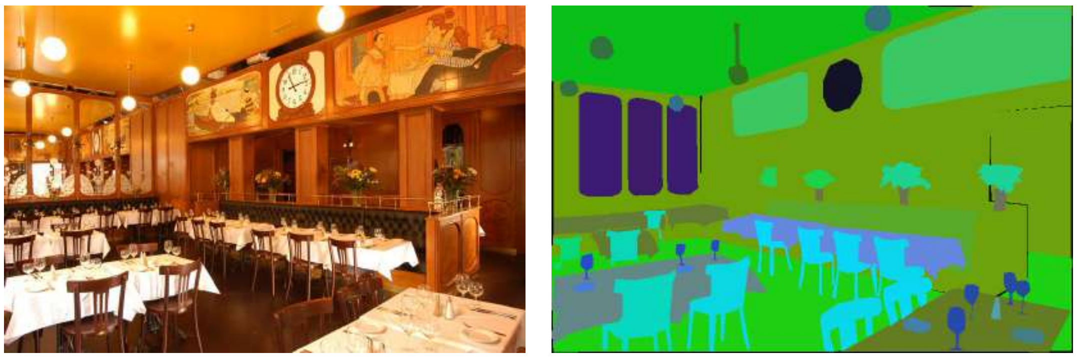
Segmentation is a task in the computer vision field that gives an understanding of what object is pictured in the image, where the object is located in the image and gives precise classification for each region of the image to what class that region corresponds. Segmentation could be divided into two subcategories: semantic segmentation and instance segmentation. Instance segmentation gives so-called segmentation masks only for objects that the algorithm was able to locate in the image. Whereas semantic segmentation, in contrast, assigns a class to each pixel in the image. However, unlike instance segmentation, semantic segmentation does not distinguish between different instances of the same object class.

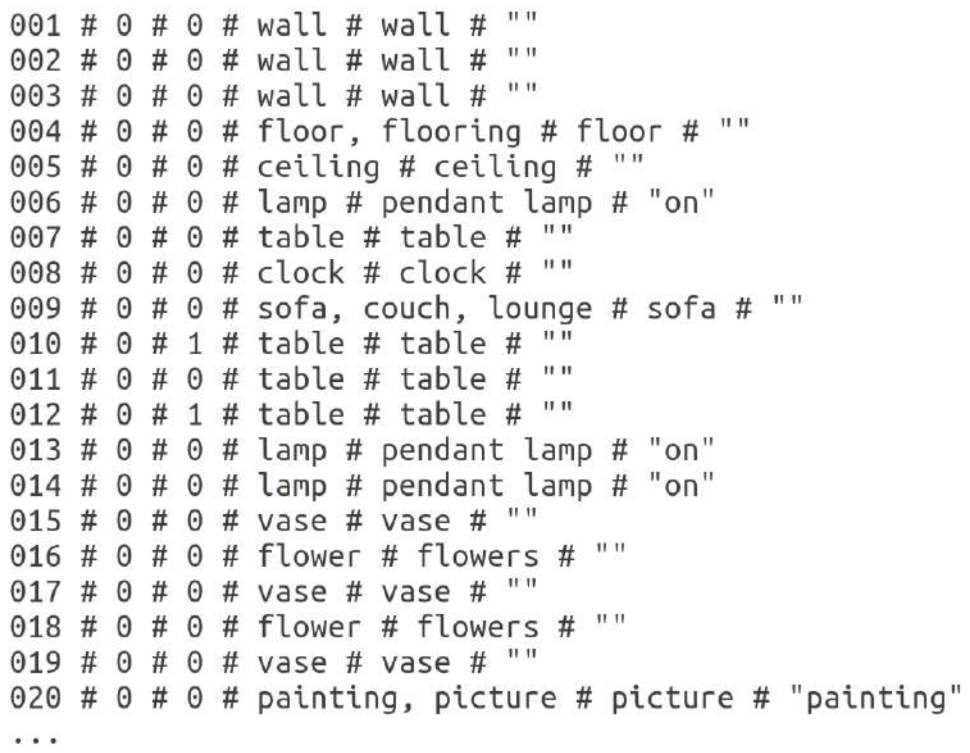
Based on the description of the task, it is desirable that the dataset for the segmentation should contain images of different scenes and corresponding segmentation masks for each image or object represented in that image. Below is a detailed description of the datasets that were used for the segmentation tasks in this thesis.

2.1.1 ADE20K

One of the datasets that were used in this work is ADE20K [Zhou et al., 2017]. It contains pictures of different scenes both outdoor and indoor. Each scene contains segmentation masks for each image, in some cases even a couple of them, providing a more detailed segmentation (see Figure 2.1 for the reference). Besides that, for each image, it contains a text file that describes the content of each image; in that way, we also have the list of classes that are present in segmentation masks in each image(Figure 2.1 C).

The advantages of this dataset are that it is quite large (20 thousand of images) and contains segmentation masks and class description for each image item. The disadvantages are that the dataset also contains outdoor scenes and the images of the scenes are separated in the sense that each image is an entirely different scene,


(A) Original image (B) Semantic segmentation



```
001 # 0 # 0 # wall # wall # ""
002 # 0 # 0 # wall # wall # ""
003 # 0 # 0 # wall # wall # ""
004 # 0 # 0 # floor, flooring # floor # ""
005 # 0 # 0 # ceiling # ceiling # ""
006 # 0 # 0 # lamp # pendant lamp # "on"
007 # 0 # 0 # table # table # ""
008 # 0 # 0 # clock # clock # ""
009 # 0 # 0 # sofa, couch, lounge # sofa # ""
010 # 0 # 1 # table # table # ""
011 # 0 # 0 # table # table # ""
012 # 0 # 1 # table # table # ""
013 # 0 # 0 # lamp # pendant lamp # "on"
014 # 0 # 0 # lamp # pendant lamp # "on"
015 # 0 # 0 # vase # vase # ""
016 # 0 # 0 # flower # flowers # ""
017 # 0 # 0 # vase # vase # ""
018 # 0 # 0 # flower # flowers # ""
019 # 0 # 0 # vase # vase # ""
020 # 0 # 0 # painting, picture # picture # "painting"
...
```

(C) List of an objects present on the image

FIGURE 2.1: An example from ADE20K dataset

so one cannot build any sequence of scenes of the rooms or corridors that connect different locations.

The objects represented in ADE20K dataset are grouped into 150 classes. All classes were divided by the authors to *staff* and *things*. To staff belongs classes like *wall*, *sky*, and *road*. To things belongs classes like *car*, *person*, and *table*. In total there are 35 classes that belong to staff and 115 classes of things [Zhou et al., 2018].

This dataset has one more disadvantage - the inconsistency of the naming of both a scene description and class labels. Namely, synonyms are used for the description of familiar scenes, which may cause complication in the evaluation phase.

The ADE20K dataset was used for pretraining the weights of CSAIL [Zhou et al., 2018] model, which is one of the models that serve as a feature generator in this thesis. The features generated by the model were applied to improve the performance of the localization. For a detailed description of a feature generation process, please refer to Chapter 3.

2.1.2 MS COCO

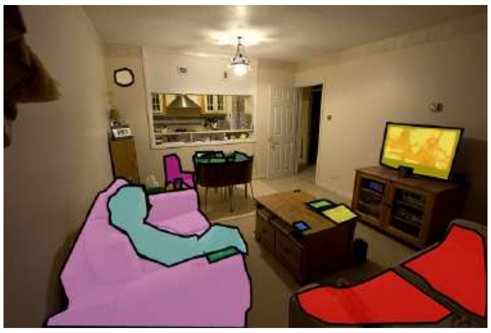
MS COCO [Lin et al., 2014] is one of the most popular datasets for different fields of computer vision. It is used in a vast variety of tasks like object segmentation or general scene understanding.

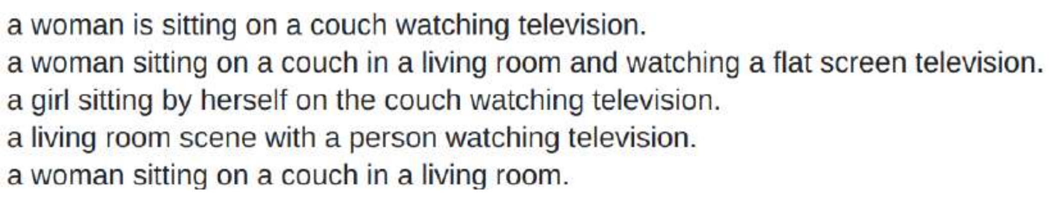
This dataset contains 330 thousand images and 1.5 million objects in the images. That means that there are five objects in each image on average. This point is critical for the problem that we are aiming to solve. The dataset that will be used for indoor localization in this thesis was gathered in a crowded university building [Taira et al., 2018b; Wijmans and Furukawa, 2017], and the MS COCO is very close to these conditions in terms of the objects represented in the images. One of the models that were used for semantic feature generation - Mask R-CNN - was used with weights pretrained on the MS COCO dataset. It is critical to have a dataset that will be good for semantic segmentation tasks.

Images in MS COCO dataset contain 80 classes of objects. The object categories were carefully chosen as most commonly used and contain general objects from common human-inhabited space. For example, classes like *kite, spoon, banana, fridge* are present. The dataset covers both indoor and outdoor environment, but due to the general nature of the dataset, it fits our needs. All classes that are critical for our purposes, like *sofa* and *chair*, are present.

MS COCO dataset consists of images together with segmentation masks for the objects that are present in that particular image. Moreover, each class that is present in an image is highlighted separately. If there are multiple instances in one image, they are highlighted together. Besides that, each image contains five sentences that describe the scene, represented in the image. Please find an example of a scene from the dataset in Figure 2.2.

(A) Original image


(B) Segmentation


a woman is sitting on a couch watching television.
a woman sitting on a couch in a living room and watching a flat screen television.
a girl sitting by herself on the couch watching television.
a living room scene with a person watching television.
a woman sitting on a couch in a living room.

(C) Scene description

FIGURE 2.2: An example from MS COCO dataset

2.2 Indoor localization dataset

The task of indoor localization is more specific than the task of segmentation, so there are more constraints for the datasets. A dataset for indoor localization is harder to collect or generate than a dataset for segmentation, for example. The primary requirement is that the images of the dataset should be bound to a specific point on the map. So, the visual data, represented by the images, are connected with some location on the map. Images cannot be taken independently from different places, as often happens in cases of more general datasets. Below is a description of the dataset that was used for the indoor localization in this work.

2.2.1 InLoc

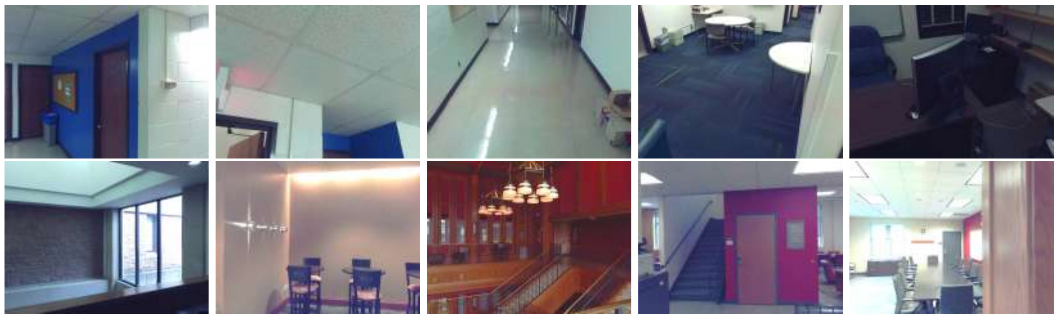
The main dataset that was used for this work is InLoc [Taira et al., 2018b]. For the purpose of future references in this work, it needs to be described in more detail.

First and the main difference of this dataset is the fact that it captures the area inside two university buildings. It is a large-scale dataset, which means it contains the layout of the entire floors of the buildings. This dataset was collected by scanning two buildings of the Washington University in St. Louis with a Faro 3D scanner [Wijmans and Furukawa, 2017].

In total, the InLoc dataset contains 277 RGBD scans that are geometrically registered to a floor plan of the mentioned buildings. All the scans are divided into five different scenes that correspond to five floors of the Washington University in St. Louis [Taira et al., 2018b]. From each scan, the corresponding perspective RGBD images were generated and that gives in total 10 thousand images [Taira et al., 2018b].

The second important thing about the InLoc dataset is the internal structure of the data. The images generated from the panoramic scans are called *database* images. They are generated from each panoramic scan by taking different angles of the viewpoint on three axes. So, the database images cover all the views of each panorama by 36 images, including floors and ceilings [Taira et al., 2018b]. Besides that, there is a relatively small dataset of 356 images called *query* images. They were captured by phone (no depth data, only RGB) in the same buildings where the panoramic scans were acquired. The main difference though is that the query images were captured significant time after the originals scans were made. That means that the illumination on query images is entirely different, some people appear in the photos and some objects (like chairs, tables, other furniture) are captured in different places. This will complicate the localization. For an example of the query and database images from the InLoc dataset, please refer to Figure 2.3.

Moreover, the query images capture only two of the five floors that are present in the database images. This will make a task of localization more complicated.


(A) Database images

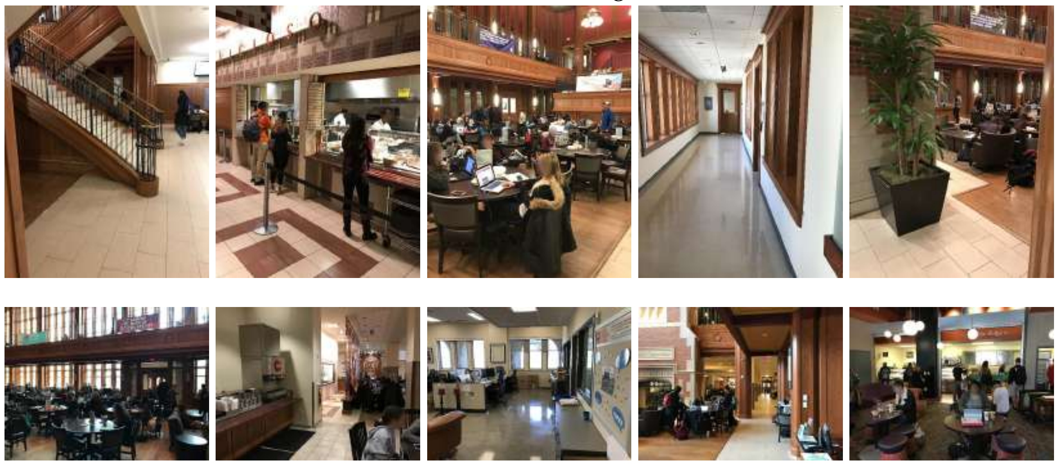

(B) Query images

FIGURE 2.3: Image examples from InLoc dataset

Chapter 3

Semantic segmentation

In this chapter the process of semantic feature generation will be presented.

As mentioned in Chapter 1, one of the main problems of the indoor localization algorithms is a dynamic environment. Many objects in the environment cannot be used as reference objects for the algorithms that are mainly relying on visual information. An example of such objects is easily movable furniture, like chairs, tables and some other objects like a houseplant or vase. Such objects could be easily moved in the environment and should not be referred as significant reference points for the localization algorithm. That brings us to the idea that there are some parts of the image that could be useful for the visual localization algorithm while other parts could instead impair the localization.

We developed this idea further and came up with a concept that semantic features could add value to the localization algorithm. In general, the idea was to use a semantic algorithm to mark out the unimportant regions on the image, and in such a way it will allow the localization algorithm to work only with the significant parts of the image.

To address this problem, we need some algorithms that could provide us with an understanding of the environment. By understanding, we mean the understanding of the physical and practical meaning of the objects that are present in the image. In other words, we need some algorithms that could provide us with the semantics of the environment.

In the field of computer vision the algorithms that give the state-of-the-art performance in semantic tasks are based on deep convolutional neural networks. We decided to try two different algorithms: Mask R-CNN and CSAIL semantic segmentation. A brief overview of the segmentation algorithms that were used in this thesis and further justification of the particular algorithms chosen will be covered in the following section.

3.1 Overview of used semantic segmentation methods

3.1.1 Mask R-CNN

Mask R-CNN [He et al., 2017] is an instance segmentation neural network. In the thesis we have used the implementation that was presented by the Matterport [Abdulla, 2018]. Mask R-CNN was introduced as an improvement of the Faster R-CNN [Ren et al., 2015; Girshick, 2015]. R-CNN stands for regional convolutional neural network. Among other improvements, instance segmentation was added in the Mask R-CNN. Instance segmentation means that in the result of the model inference on this image it will mask out all the objects that were found in it. Moreover, the model will distinguish the objects of the same class and mark them as different instances of the same object. For the difference in the segmentation approaches, please refer to the Chapter 2.1.

Mask R-CNN makes a prediction in two steps. In the first step, it goes through the image and generates so-called proposals - parts of the image that might contain objects. In the second step, the model classifies the proposal, and if it contains a valid object, draws a bounding box and a segmentation mask.

Now we will describe the architecture of the Mask R-CNN in detail. There are the following modules inside the Mask R-CNN architecture.

1. **Backbone**

   It is a term that is often used to name the feature extractor. In the realization of the Mask R-CNN that was used in this thesis [Abdulla, 2017] the backbone is based on ResNet101 architecture [He et al., 2015]. As a typical convolutional feature extractor, first layers of the backbone are responsible for the simple features, like lines and corners, and the last layers are standing for the more complex features, like objects and more complex structures.

   In the next step, the existing feature map is improved by the Feature Pyramid Network (FPN) [Lin et al., 2016]. This allows the object to be better represented on different scales.

   The feature map generated by the backbone is the input of the next module.

2. **Region Proposal Network (RPN)**

   RPN is a network that scans the image with a sliding window trying to find the areas with the objects. The regions that RPN scans over are called anchors. However, in a real case, the RPN is producing over 200 thousands anchors of different ratio and scale, so they cover all the image. Also, the RPN is running over the feature map, not an original image. This allows to avoid the same calculation twice and affects the performance. Moreover, as RPN is a convolutional network, so the scanning process could be paralleled.

   For each anchor RNP generates two outputs:

   (a) **Anchor class**
   This field contains one of two values: background class or foreground class. Foreground indicates that there is an object in this particular anchor.

   (b) **Bounding box refinement** Anchor rarely fits the object perfectly, so RNP generates estimation on how to refine the position and the shape of the anchor to fit the object perfectly.

   With the RPN output, the top anchors are chosen. If some anchors are overlapping, the one with the highest foreground score is chosen. That creates the region of interest (ROI) that goes to the next stage.

3. **ROI classifier and bounding box regressor** This part is running over the ROI proposed by RPN and similarly to RPN generates two types of output.

   (a) **Class** The neural network that is used on this stage is a classifier that can predict a specific object class like a *bird*, *cake* or *bed*. However, this classifier can also predict class foreground, which means that there is no object in this ROI and it should be discarded.

   (b) **Bounding box refinement** Similarly to the RPN case, this allows tuning the position of the object bounding box in the image.

4. **Segmentation mask** In parallel, the convolutional network is used to generate the segmentation mask based on the tuned ROI from the previous stage.

The Mask R-CNN has three main outputs. First one is the class label for each instance of the object. The label model also returns the probability for every predicted label. The second is a bounding box for every object that model was capable of detecting on the image. The third output is a segmentation mask for every object inside the bounding box. For an example of the output, please refer to Figure 3.1.

3.1.2 CSAIL

The CSAIL is deep convolutional neural network architecture that was proposed for the semantic segmentation on the ADE20K scene parsing dataset.

The idea behind the CSAIL approach is to stack existing deep convolutional neural networks. The high-level architecture of the model is similar to the autoencoder. For this thesis we used the dilated ResNet50 [He et al., 2015], and for the decoder we used the Pyramid Pooling Module from the PSPNet architecture [Zhao et al., 2017].

The output that is produced by this architecture is the segmentation mask. Segmentation mask is the image of the same size, as the input image, but all the pixels from the original image are representing the class of the object, that was in the original image. In the segmentation mask regions that represent the same class is of the same color. For the example of the CSAIL semantic segmentation, please refer to Section 3.3.1.

3.2 Mask R-CNN for object segmentation

In this chapter, we will cover different approaches that were tested with the application of Mask R-CNN [He et al., 2017].

This architecture is widely used across different fields that require a semantic understanding of the environment. The Mask R-CNN has a proven capability to handle complex segmentation tasks from autonomous driving to factory process overview and cell tracking on microscopy images [Abdulla, 2017]. So, we decided to try this architecture for the segmentation in the indoor environment.

However, the problem was with the datasets. As we have mentioned previously, InLoc dataset was collected with the goal to be used for localization, not semantic segmentation. That implies that the dataset contains only RGBD data and no labels for the segmentation. So, we cannot start training Mask R-CNN on the InLoc dataset as is.

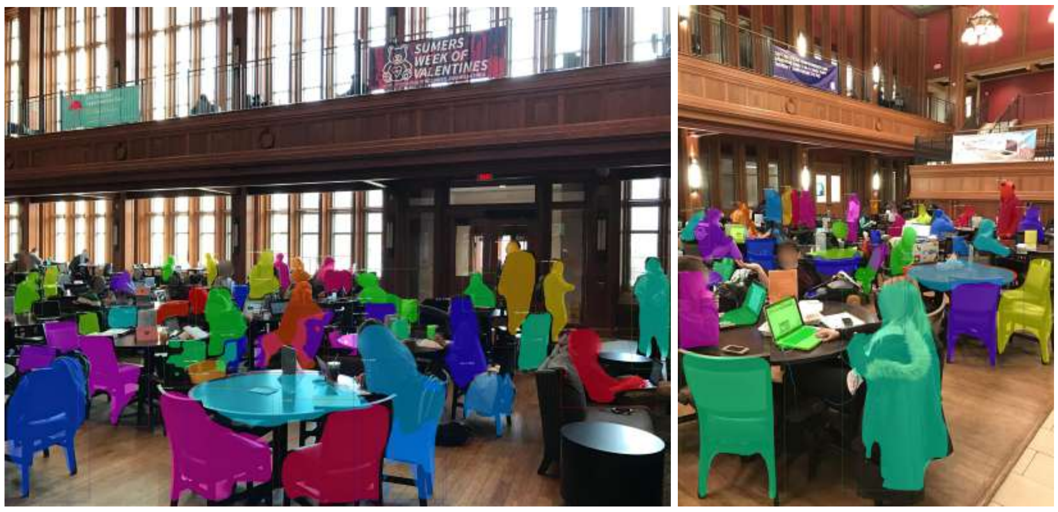
Moreover, InLoc dataset is a big dataset for localization task. However, for training a deep neural network for segmentation from scratch, 10 thousand images are not enough to achieve good results.

Taking that all into consideration and with the lack of the resources and time to label InLoc dataset, we decided to use Mask R-CNN with pretrained weights.

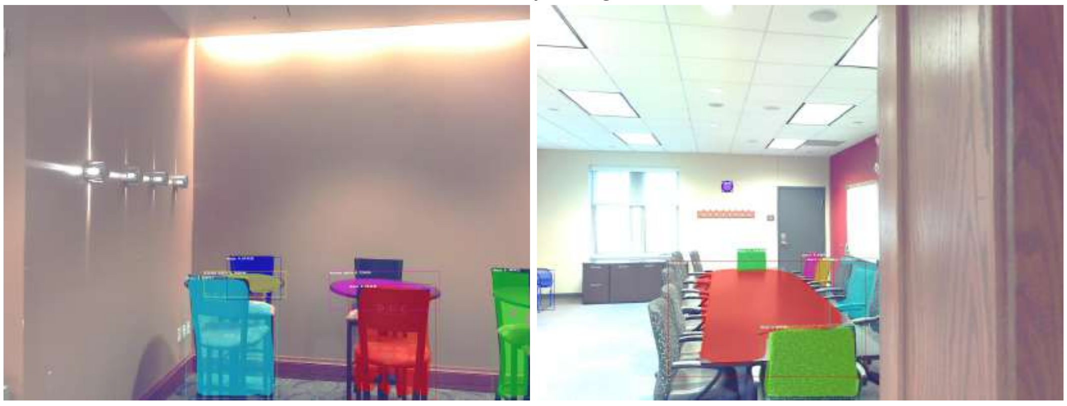
We decided to use weights pretrained on the MS COCO dataset(please refer to the Section 2.1.2). The reason for that was the presence of more complex scenes represented in MS COCO than in ImageNet, for example. The scenes represented in COCO contain much more object instances in one scene. In that way the dataset is more similar to the natural environment and the environment captured in the InLoc [Taira et al., 2018b]. This fact increases robustness of the model.

COCO dataset contains 80 object categories. Taking into account the nature of the indoor environment and the objects represented in that environment, the overlap of these two categories(classes of COCO and objects represented in InLoc) gives us exact type of the results, that we have expected. Generated segmentation masks contain objects that are obsolete for indoor localization.

The results of Mask R-CNN inference with pretrained MS COCO weights are presented in Figure 3.1.



(A) Query images



(B) Database images

FIGURE 3.1: Mask R-CNN inference results

As we can see, the classes of COCO and the objects represented in InLoc allow us to mark "movable" objects on the scenes. The classes that are segmented in Figure 3.1 are mainly chairs and people with rare exceptions. The boundaries of the segmented objects are not clear, and there are objects that were not segmented - for example, chairs on left query image and right database image on Figure 3.1.

Strictly speaking, the intersection of the COCO classes and the objects present in the InLoc dataset is giving the exact type of the results that are needed: mark out the parts of the image that obstruct the localization pipeline.

3.2.1 Binary masks

We decided that for the beginning it would be enough to have a binary mask for every image in order to distinguish “movable” objects and not take them into account for further calculations in the pipeline.

The binary masks were calculated this way. First, we run the inference of Mask R-CNN with COCO-pretrained weights on all images from the InLoc dataset (both query and database images). After that we run a custom script that does the following:

```
Data: Image, processed by the Mask R-CNN
Result: binary mask corresponding to the input image

foreach pixel in image do
    read pixel;
    if pixel is part of the mask then
        pixel value = (255, 255, 255);
    else
        pixel value = (0, 0, 0);
    end
end
```

Algorithm 1: Transfering the output of the Mask R-CNN to the binary mask

So, we marked with white (255, 255, 255) the part of the image that contains the objects detected by the Mask R-CNN. Other parts of the image remain black (0, 0, 0). Examples of the binary masks for both query and database images are presented in Figure 3.2.

The masks generated by algorithm 1 are containing less data than an output of Mask R-CNN, but they have a relatively smaller size. The size optimization is the change comparing to the original Mask R-CNN output. The binary mask still has the same problems as the original Mask R-CNN output: imprecise boundaries and unsegmented objects.

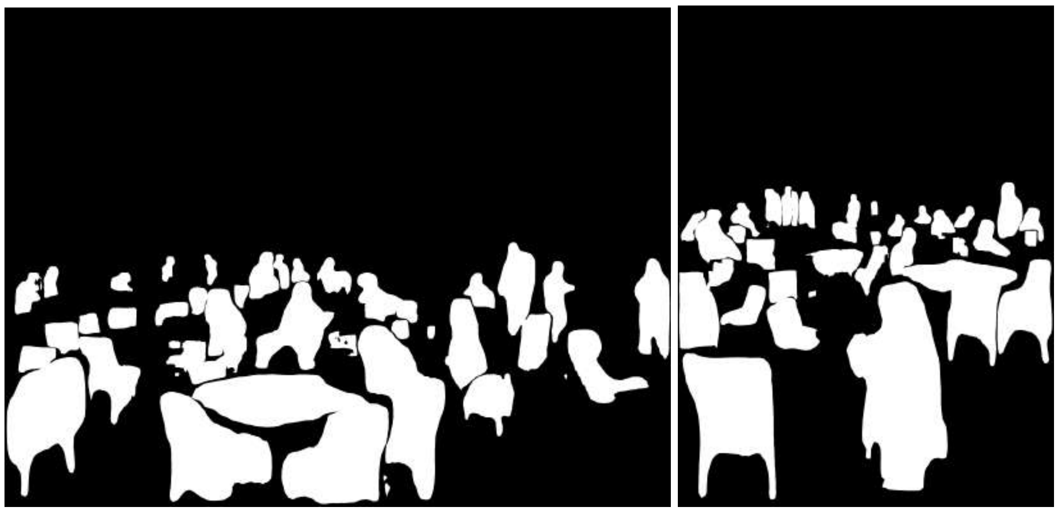
After generation, these images were added to the pipeline. The mask was used to mask out the parts of the image that need to be omitted for further calculations. This allows the algorithm to work only with the parts of the image that contains only objects that will be useful for the localization, like walls, ceiling, and floor.

One more thing worth mentioning regarding the binary masks is that they are very computationally efficient. On average, the size of the binary mask is 10 Kb for a database image and 60 Kb for a query image. This size of the image and the fact that it is black-white colored allow this solution to be highly computationally efficient, as the image could be represented as a binary matrix.

In terms of the algorithm performance that was brought by the binary masks, they did not improve the pipeline. That brings us to our next idea.

3.2.2 Instance mask

In the implementation of the Mask R-CNN that we have used [Abdulla, 2017], the colors for the resulting segmenting mask are picked randomly. However, each segmentation mask is surrounded by the bounding box, with the confidence score and


(A) Query images

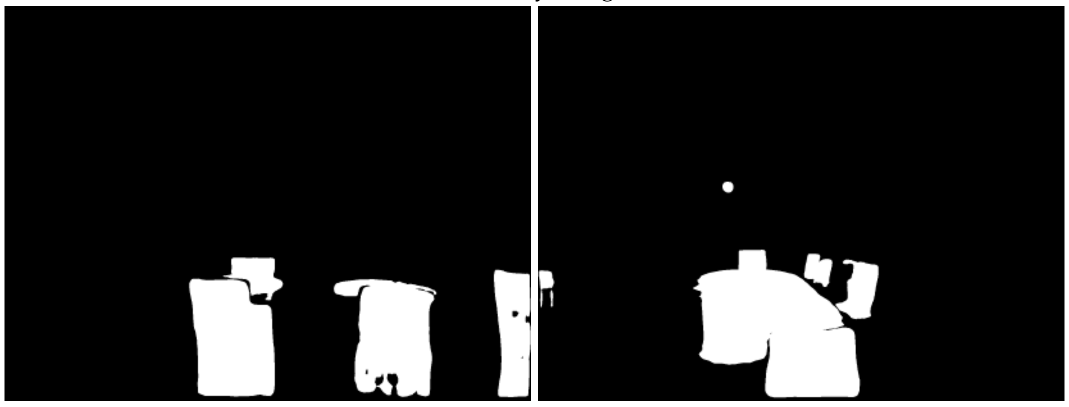

(B) Database images

FIGURE 3.2: Mask R-CNN binary mask results

label of the object in that specific mask (please see Figure 3.1). But, as could be seen in Section 3.2.1, we remove all the extra info that was not required by the algorithm.

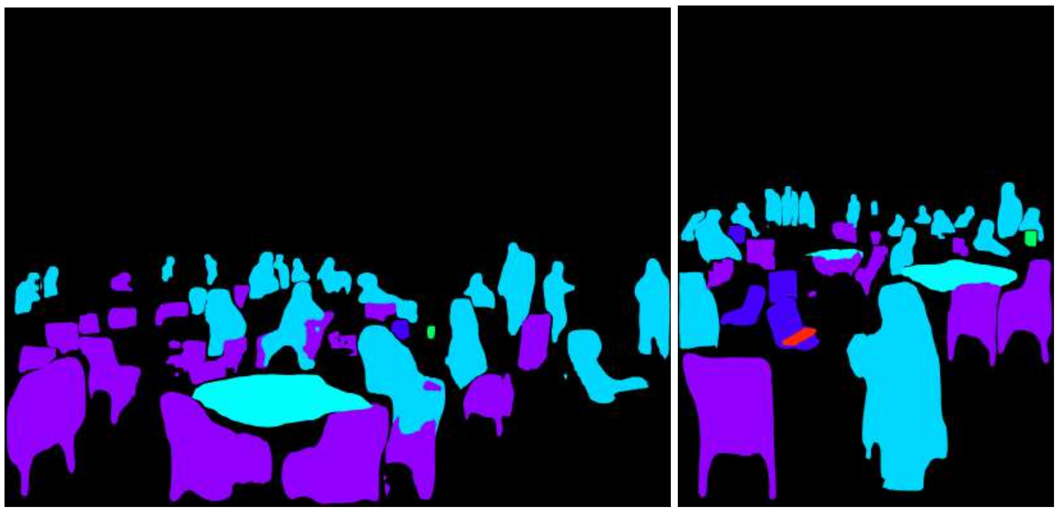
A random color pickup was not good for our task, because in that case objects that belong to the same class are marked with different colors on different images, and even the same object might have different colors if we run the inference on the same image once again.

We came up with a solution and define colors for each class of COCO by ourselves. This allows us to have the different instances in different images to be segmented with the same color.

The result of the modified inference of the Mask R-CNN with the class-wise object segmentation could be found on Figure 3.3. The improvement, compared to the binary masks, is that now we can filter out some classes and leave the rest on the image. For example, removing people (marked blue) and chairs (marked purple) will help to bring more clear predictions.

3.3 CSAIL for the semantic segmentation

In this chapter, we will describe the results that were generated with the architecture CSAIL.


(A) Query images

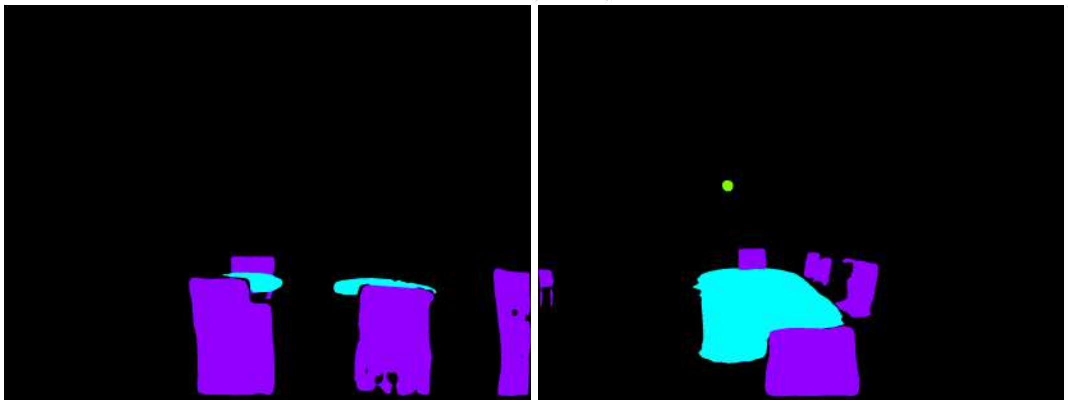

(B) Database images

FIGURE 3.3: Mask R-CNN color mask results

CSAIL is a shortcut name of the PyTorch implementation of semantic segmentation models on the ADE20K dataset (please refer to Section 2.1.1 for the details) [Zhou et al., 2017]. The exact architecture that was used in this work is ResNet-50 [He et al., 2015]. Also, as it was with Mask R-CNN in Section 3.2, we used pretrained weights. However, this time the model was trained on the ADE20K dataset, which was designed specifically for the scene parsing, which implies that a significant part of the dataset contains the indoor scenes.

The idea behind using the semantic segmentation for this task is that in the output of the CSAIL model we will have each pixel of the input image assigned to some specific class. So, we will receive not only the object segmentation masks, like in the case of Mask R-CNN. Besides that, we also receive segmentation of, e.g. doors and windows, as classes that represent them were absent in the COCO dataset. Moreover, in the case of CSAIL, we also have classes like *floor, wall, ceiling*.

3.3.1 Color segmentation

The first type of results that were generated by CSAIL is semantic segmentation. For that, we use weights pretrained on ADE20K and ResNet-50 architecture.

An example of the results is listed below. The examples are presented in Figure 3.4 with the corresponding original image to have a better understanding of the results. The results that we have from CSAIL are better than the results from the Mask R-CNN. The advantage of semantic segmentation is that we have segmented all the objects in the image, unlike in the case of Mask R-CNN. Also, we have all the important classes recognized in the image: *chairs, people, floor, ceiling,* etc. This shows the advantage of the ADE20K dataset, that was used for pretraining weights of CSAIL. One problem left unresolved is the segmentation of boundaries, which are still not clear.

One more important thing to mention is that in the case of the ADE20K each class has it is color defined by the dataset creators, so there was no need to generate colors for these classes.

3.3.2 Grayscale masks

Next thing that we have tied to achieve is to have the masks from CSAIL in grayscale.

We have modified the way how CSAIL model produces the output and make it return the same predictions but in grayscale. The advantage of this approach is the memory optimization. The color mask size on average was 30 Kb big, and the grayscale correspondence was three times smaller - 10 Kb. It may look not so big in terms of one image. However, if we take into consideration that the dataset contains 10 thousands of images and these images should be used during the evaluation of the query image, the conversion to grayscale makes sense.

From the technical side, grayscale images were relatively easy to achieve. Originally, the CSAIL returns the number of the class that it predicts for each pixel of the input image. After that these numbers were interchanged with the corresponding colors for each class number. So, what we have done is encode the number of the class as a grayscale value of that class. The results could be seen in Figure 3.5. The grayscale masks are another representation of the colored masks, so there is no improvement besides space optimization.

In some places, it is hard to visually distinguish the grayscale values, especially when we have the boundary of the segments of the classes that have consecutive numbers. However, for the algorithm, it is easily distinguishable.

3.3.3 Probability masks

For some parts of the localization pipeline, it is crucial to have as precise boundaries of the objects as possible (for details about the pipeline, please refer to Chapter 4).

So far, the data contain nothing regarding the probability of that result. However, the CSAIL model calculates also info about the probability of the class prediction for each pixel of the image. We decided to have that data as the precision of the prediction. For a better understanding of how the output looks like, please find the examples in Figure 3.6.

As it could be seen, the result is a grayscale image. We want to emphasize that the probability represented on the images is the probability of the class that was chosen by the predictor as the best prediction.

The white regions on the probability mask indicate that part of the mask belonging to the predicted class with high confidence. On the other side, the darker the part of the mask, the lower the probability of that part of the mask belonging to the predicted class. As can be seen on the examples above, the dark parts of the images are concentrated near the boundaries of the objects. Having that data, we would be able to put our threshold on the boundaries probability and consider only parts with high probability.

3.4 Grouping of COCO and ADE classes for InLoc

3.4.1 MS COCO

With the class segmentation, it is easy to mark out all instances of a particular class — for example, all people or all chairs. That was an important feature for our pipeline. However, from another perspective, sometimes it is hard to decide what class of objects is more important for the localization. Taking into consideration that the decision of the class importance should be made 80 times, we decided to cluster all the classes into six different groups: *people, transient, stable, fixed, outdoor, background*.

Cluster *people* contains only one class : *person*. Cluster *transient* contains mainly small furniture like *chair* and small object classes like *book, vase, scissors* that are easy and likely to be moved. Cluster *stable* contains mainly big furniture and objects that are unlikely to change their position, for example, *bed, dining table, tv*. The purpose of the cluster *fixed* is to contain objects that will probably never change their position in space, like *doors* or *window* for example. Cluster *outdoor* was meant to contain all objects that will most likely never be present in indoor scenes, like *car, airplane* and *boat*. The cluster *background* is rather technical because the purpose of this class is to contain all parts of the image that are not covered with segmentation masks, i.e., the region that contains the most valuable information for further calculations. As we define our color map for all the classes in COCO dataset, a color that corresponds to the *background* cluster is black. All the COCO classes, clusters proposed by us can be found in Appendix A.

3.4.2 ADE20K

The number of classes in the ADE20K was even higher than in the COCO dataset, so we decided to cluster all the classes in the ADE dataset to the same categories we use in the COCO case. Namely: *people, transient, stable, fixed, outdoor, background*. This solution allows us to work consistently with the classes of the different datasets - MS COCO and ADE20K. For the purpose of our task, the name of the class is not crucial as far as that class belongs to the same category. The full list of the ADE20K classes and their clustering can be found in the Appendix B.

There is two main difference for the ADE dataset that we want to highlight here. Firstly, in case of clustering of the ADE, we have more classes in the category *stable* and *fixed*. It is not only due to the higher number of classes in general, comparing to the COCO, but also (and mainly) because ADE is more specialized for the indoor environment. To be specific - ADE has objects like *wall, stairs,* and *fireplace* that were defined as part of *fixed* category, as well as objects like *wardrobe* and *coffee table* that were put into the *stable* category.

Secondly, there are inconsistencies in the names of the classes of the ADE. For example, we have both classes *sofa* and *couch*.The words are basically synonyms, and there are no clear rules on how to distinguish the sofa and couch visually. So, regarding the existing segmentation model, it has an internal bias that comes from the labeling for the distinction of the sofa and couch. However, concerning our task, it is not important whether it is a sofa or a couch - it is crucial that it is the object from the category *stable*.

The last thing to mention about the difference between clustering of COCO and ADE is that CSAIL is a model for the semantic segmentation, which means that each pixel in the image will be assigned to some class. Because of that, the cluster background makes no sense, so we left it empty.

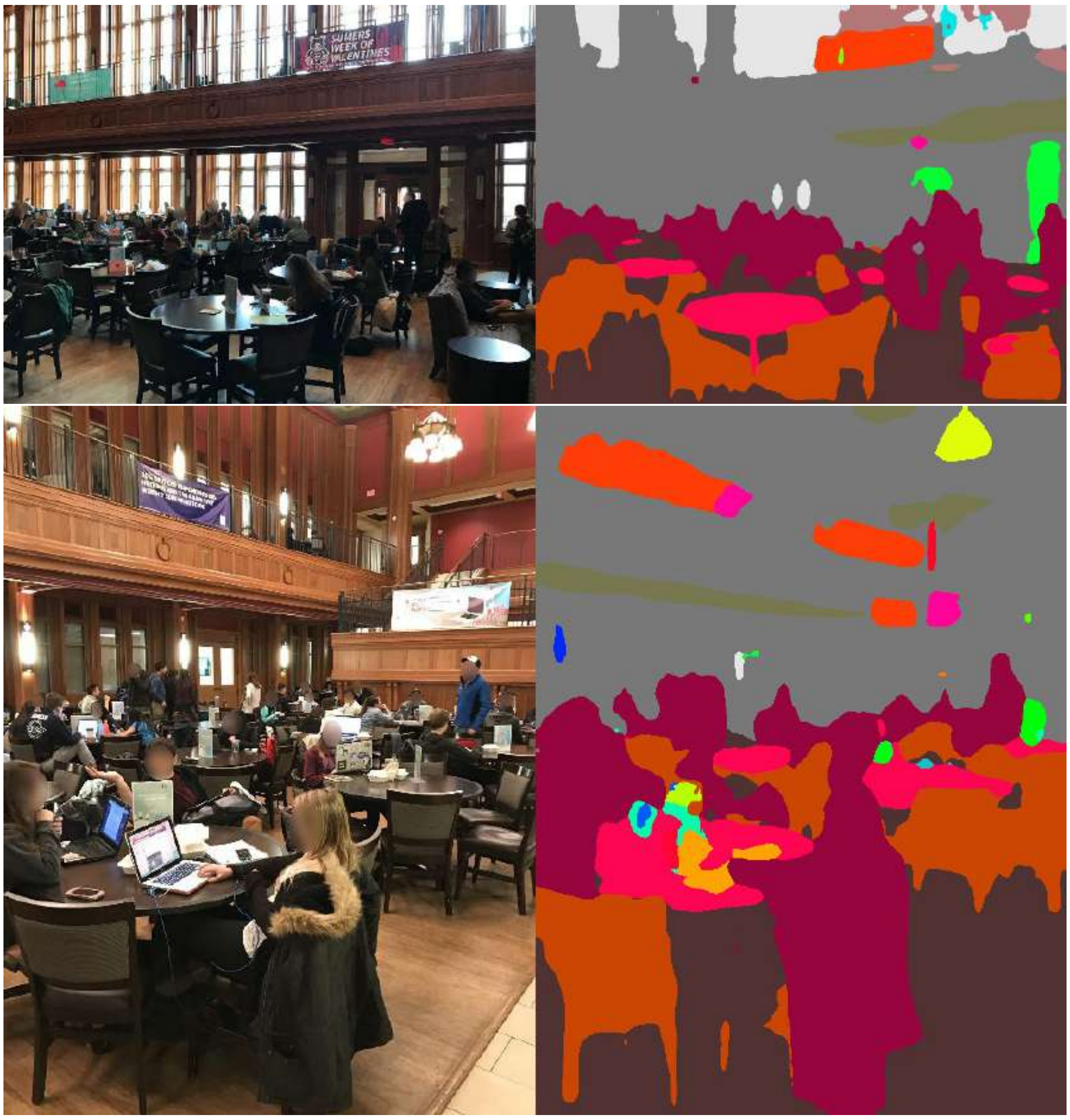
The exact mapping of the classes of the ADE and the corresponding clusters can be found in Appendix B.

3.5 Segmentation evaluation

After a qualitative evaluation we came to the conclusion that the masks generated by the CSAIL will give more benefits for the pipeline performance. The output produced by the CSAIL gives segmentation results that are more suitable for the indoor localization. Unlike the Mask R-CNN, CSAIL segments all the image, which is critical for the performance of the localization. The list of classes that CSAIL is able to segment is also bigger and more suitable for the indoor environment.

The better performance of the CSAIL could be explained by the fact that CSAIL was pretrained on the dataset (ADE20K) that better fits the domain. For example, ADE20K contain classes like *wall*, *ceiling* and *floor* which are more useful for the indoor environment.

For the final evaluation of the indoor localization performance in Chapter 4 we used masks generated by the CSAIL semantic segmentation.


(A) Query images

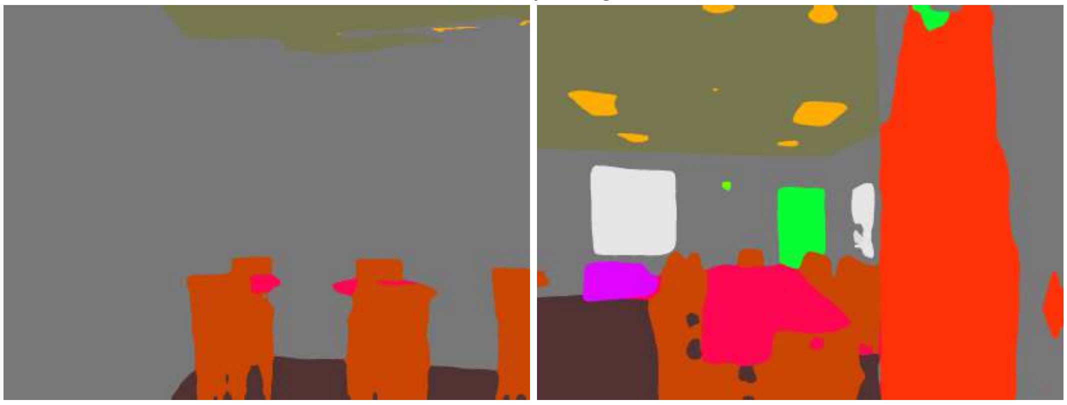
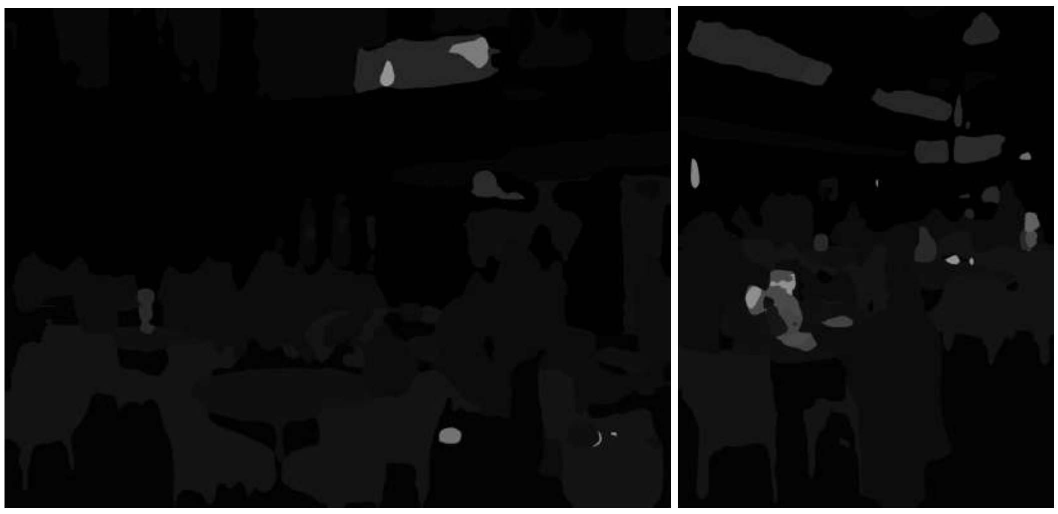
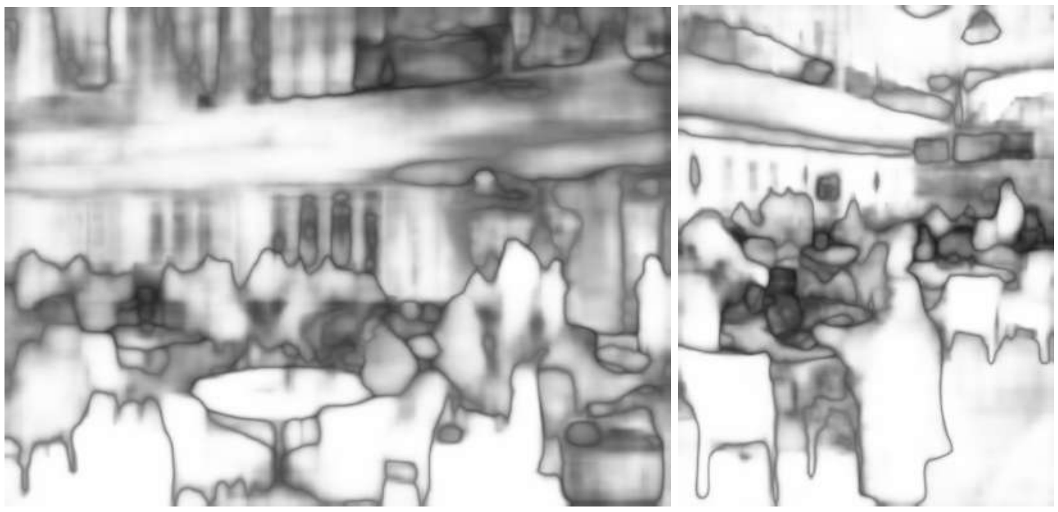

(B) Database images

FIGURE 3.4: CSAIL color mask results


(A) Query images

(B) Database images

FIGURE 3.5: CSAIL grayscale mask results


(A) Query images

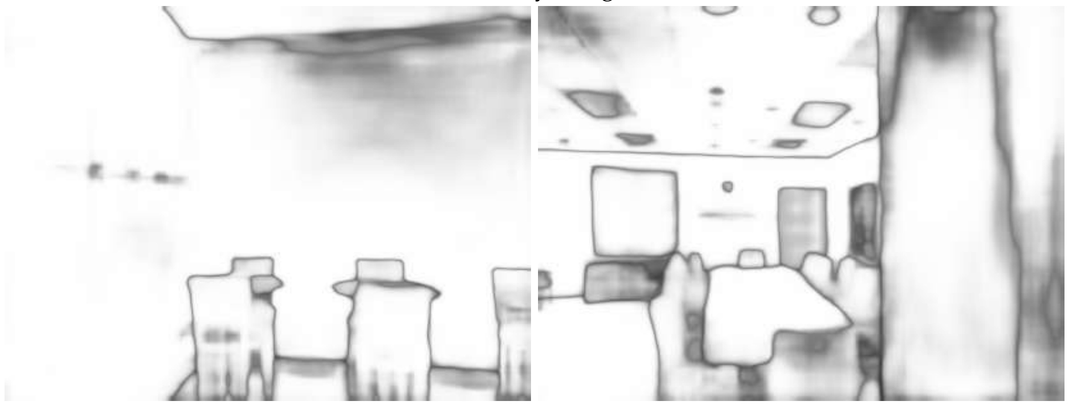

(B) Database images

FIGURE 3.6: CSAIL probability mask results

Chapter 4

Indoor visual localization with semantic segmentation

4.1 Overview of InLoc approach

InLoc is a state-of-the-art approach for indoor localization [Taira et al., 2018a] exists a pipeline for predicting the location of a query image on the 3D map that was built from the 3D scans (please refer to the Section 2.2.1 for the details). For future reference and to give better understanding and motivation of our contribution, we will describe the pipeline in details.

The pipeline consists of three steps. The first step is called candidate pose retrieval. The purpose of this step is to get the best N candidate images from the database images. It is achieved by generating dense features from the architecture NetVLAD [Arandjelovic et al., 2015]. Dense features are generated for both query and database images. After that the normalized L2 loss is computed. Based on the loss, at the end of the first step, there are top N pose candidates [Taira et al., 2018b].

The second step is called pose estimation (PE). In this step, features extracted from the convolutional neural network were used (VGG-16 in this particular case). The idea is to find a geometrically consistent set of correspondences. It allows to describe more higher-level information with a larger receptive field. Taking into consideration that database images contain depth information, the query image correspondence could be estimated by pixel-to-pixel correspondence between the query and database images [Taira et al., 2018b].

The third step is called pose verification (PV). Using the precise data in the database RGBD images, a virtual 3D view from the pose estimated in the previous step is rendered. This 3D view is compared with the real view of the query image. Based on that comparison, the evidence of the query image belonging to that particular scene is calculated. One of the novel parts presented in the InLoc is the way of calculation of the evidence of similarity between the images. In the InLoc both positive and negative similarity is calculated. It is done in a pixel-wise manner over the regions that are and are not consistent between the query image and the generated 3D view [Taira et al., 2018b].

The work that is the subject of this thesis is dedicated to the second and the third step of the pipeline - pose estimation (PE) and pose verification (PV) respectively.

4.2 Main results

The semantic segmentation features, generated in Chapter 3, were integrated into the InLoc pipeline. Figure 4.4 illustrates how the semantic features were used in the InLoc pipeline.

The main results are presented in Figure 4.3. These results were obtained and kindly provided by Hajime Taira at Tokyo Institute of Technology.

The segmentation masks obtained in Chapter 3 were applied to different parts of the pipeline. In this chapter we will discuss the results that were obtained by applying the segmentation masks to the pose estimation and pose verification parts of the pipeline.

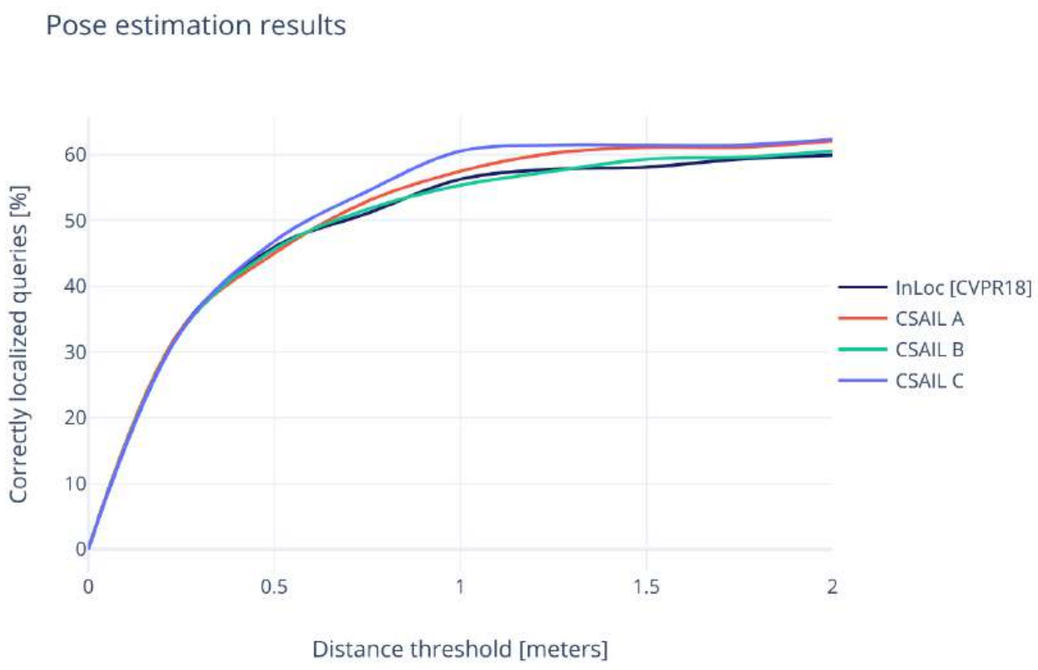


FIGURE 4.1: Performance of the InLoc pipeline with semantic features added to the pose estimation step

Figure 4.1 presents the performance of the InLoc pipeline with added semantic features on the pose estimation step. The plot shows the ratio of correctly localized queries within a certain distance. The distance, with a step of 0.25m, is on the horizontal axis and the percentage of correctly localized queries is on the vertical axis. There are four curves on the plot. The black curve marks the baseline of the InLoc approach. There are three different CSAIL curves, marked A, B, and C. The difference between them are the class groups used for creating the CSAIL binary masks. For details about classes grouping, please refer to Section 3.4.

There are different CSAIL approaches represented in the plot. They are different in terms of applying segmentation masks to clusters as described in Section 3.4. CSAIL A masks contain only *stable* and *fided* clusters. The curve for this approach is red. CSAIL B masks contain all the clusters except for *people* and the corresponding curve is green. CSAIL C approach contains all the clusters delete without *people* and *transient*. The curve for CSAIL C is marked blue.

So, we have tried two different approaches here. We tried to add the masks that contain only the stable parts of the image (CSAIL A) or to remove just movable clusters and leave everything else in place (CSAIL B, C). As we may see from the plots, CSAIL C performs better than the other approaches. Starting from the distance 0.5m, CSAIL C performs better than all the other approaches. The most significant performance improvement is for distance 1m - CSAIL C performs 4.26 percentage points better than the original InLoc pipeline. For distance 2m CSAIL A and C both perform almost three percentage points better than the original pipeline.

In Figure 4.2 we have the performance of the InLoc pipeline with semantic features on pose estimation and pose verification steps.

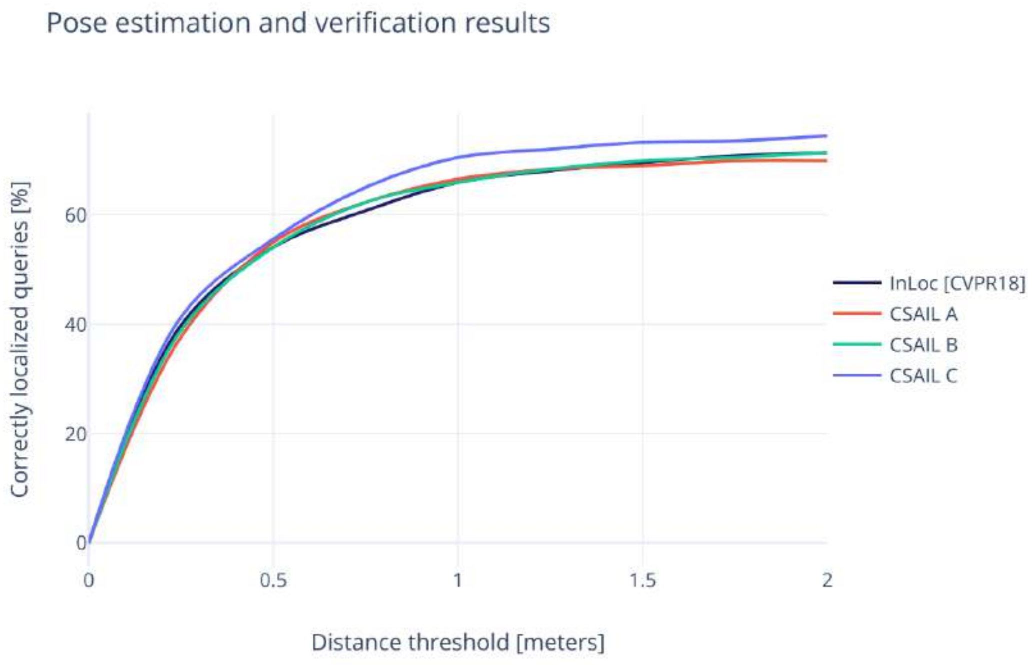


FIGURE 4.2: Performance of the InLoc pipeline with semantic features added to the pose estimation and pose verification step

As we can see in Figure 4.2, the CSAIL C approach outperforms all the other methods starting from the distance 0.6m. CSAIL C is better by three percentage points in average for the distance 0.75-2m. The other two CSAIL approaches have the performance on the level of the original InLoc.

Figure 4.3 presents a performance of all the approaches that we have tried. There are two groups of curves. One shows the overall performance of the pipeline if the semantic features were added only to the pose estimation (PE) step. The other group shows the results of adding semantic CSAIL features to both pose estimation and verification steps (PE + PV). The PE + PV approach performs better than the PE by 7percentage points in average in distance 1-2m. From this plot we can infer that the CSAIL C approach outperforms the original InLoc for both PE and PE + PV cases.

Figure 4.4 shows an example of improvement in InLoc localization obtained by use of semantic features (CSAIL C) in PE + PV.

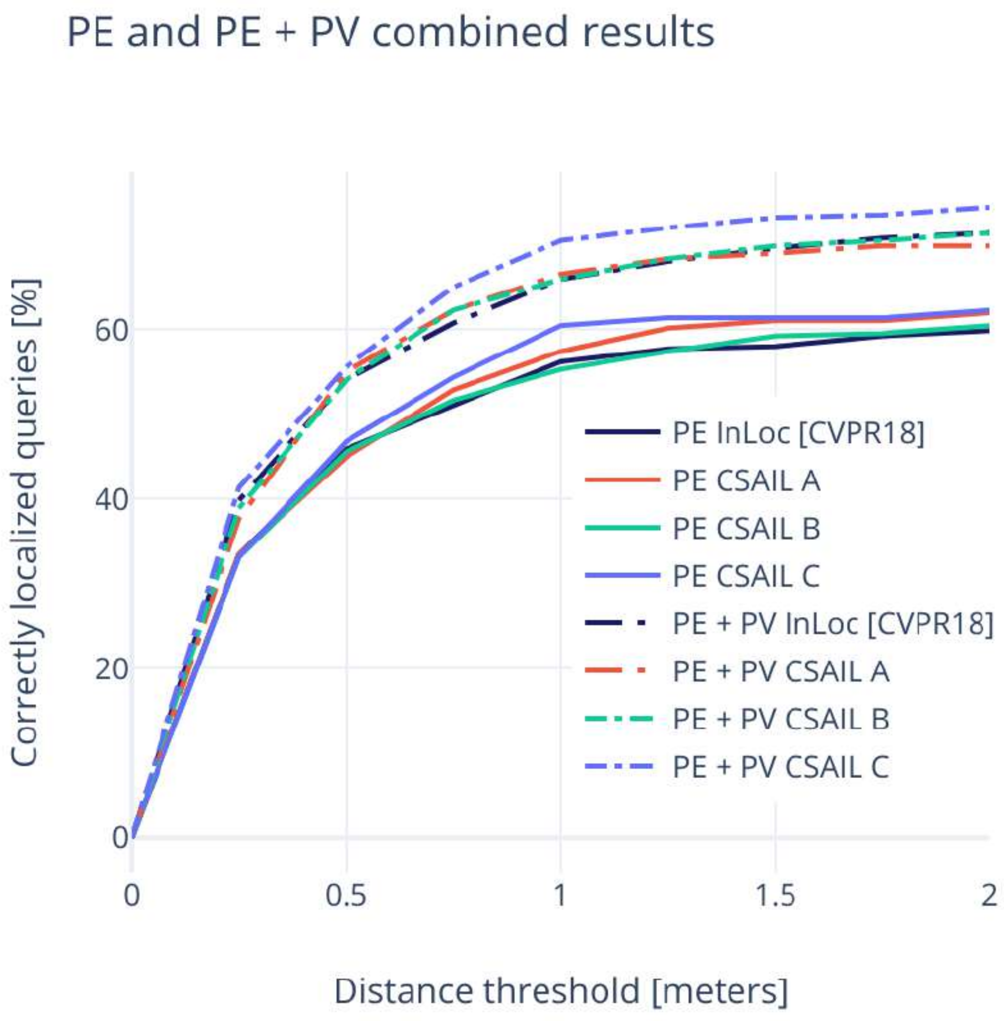


FIGURE 4.3: Performance of the InLoc pipeline with all proposed approaches

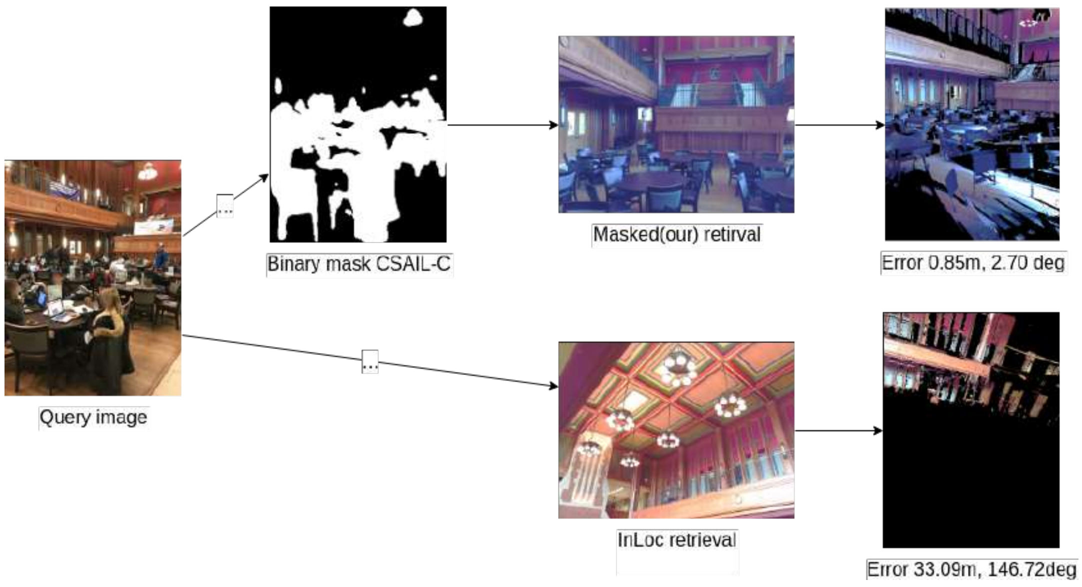


FIGURE 4.4: Example when semantic features helped to predict the correct location

Chapter 5

Conclusions and perspectives

We have presented new semantic features that improve the performance of the existing InLoc pipeline for indoor localization.

For future research in this field, the most promising part of work is an improvement of segmentation boundaries precision [Harley, Derpanis, and Kokkinos, 2017]. We have confidence based on our results that this is the part that could bring the most valuable improvements to the pipeline. For example, having more precise boundaries of the walls and other stable objects would help in the pose verification step.

Another area for improvement is prediction of the normals of the surfaces in the image. This would provide some underlying geometry and 3D structure of the environment [Bansal, Russell, and Gupta, 2016]. Taskonomy [Zamir et al., 2018] is a model that allows calculation of surface normals, as well as edges and other results with good potential for further improvement of the localization pipeline. For more details about the Taskonomy outputs and how we want to apply them in the pipeline, please refer to Appendix C.

Appendix A

′people′: person,

′transient′: bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe, backpack, umbrella, handbag, tie, suitcase, baseball bat, baseball glove, skateboard, tennis racket, bottle, wine glass, cup, fork, knife, spoon, bowl, banana, apple, sandwich, orange, broccoli, carrot, hot dog, pizza, donut, cake, chair, potted plant, laptop, mouse, remote, keyboard, cell phone, toaster, book, vase, scissors, teddy bear, hair drier, toothbrush

′stable′: bench, couch, bed, dining table, tv, microwave, oven, refrigerator, clock,

′fixed′: toilet, sink

′outdoor′: bicycle, car, motorcycle, airplane, bus, train, truck, boat, traffic light, fire hydrant, stop sign, parking meter, frisbee, skis, snowboard, sports ball, kite, surfboard,

′background′: BG

Appendix B

'people': person; individual; someone; somebody; mortal; soul

'transient':plant; flora; plant; life, curtain; drape; drapery; mantle; pall, chair, mirror, rug; carpet; carpeting, armchair, seat, desk, lamp, cushion, base; pedestal; stand, box , grandstand; covered; stand, case; display; case; showcase; vitrine, pillow, screen; door; screen, flower, book, computer; computing; machine; computing; device; data; processor; electronic; computer; information; processing; system, swivel; chair, hovel; hut; hutch; shack; shanty, towel, apparel; wearing; apparel; dress; clothes, ottoman; pouf; pouffe; puff; hassock, bottle, plaything; toy, stool, barrel; cask, basket; handbasket, bag, cradle, oven, ball, food; solid; food, trade; name; brand; name; brand; marque, pot; flowerpot, animal; animate; being; beast; brute; creature; fauna, bicycle; bike; wheel; cycle, screen; silver; screen; projection; screen,

blanket; cover, sconce, vase, tray, ashcan; trash; can; garbage; can; wastebin; ash; bin; ash-bin; ashbin; dustbin; trash; barrel; trash; bin, fan, plate, monitor; monitoring; device, radiator, glass; drinking; glass,

'stable':bed, cabinet, table, painting; picture, sofa; couch; lounge, shelf, wardrobe; closet; press, bathtub; bathing; tub; bath; tub, chest; of; drawers; chest; bureau; dresser, refrigerator; icebox, pool; table; billiard; table; snooker; table, bookcase, coffee; table; cocktail; table, bench, countertop, stove; kitchen; stove; range; kitchen; range; cooking; stove, arcade; machine, television; television; receiver; television; set; tv; tv; set; idiot; box; boob; tube; telly; goggle; box, poster; posting; placard; notice; bill; card, canopy, washer; automatic; washer; washing; machine, oven, microwave; microwave; oven, dishwasher; dish; washer; dishwashing; machine, sculpture, shower, clock

'fixed':wall, floor; flooring, ceiling, windowpane; window, door; double; door, railing; rail, column; pillar, sink, fireplace; hearth; open; fireplace, stairs; steps, stairway; staircase, toilet; can; commode; crapper; pot; potty; stool; throne, chandelier; pendant; pendent, bannister; banister; balustrade; balusters; handrail, escalator; moving; staircase; moving; stairway, buffet; counter; sideboard, stage, conveyer; belt; conveyor; belt; conveyer; conveyor; transporter, swimming; pool; swimming; bath; natatorium, step; stair, bulletin; board; notice; board,

'outdoor':building; edifice, sky, tree, road; route, grass, sidewalk; pavement, earth; ground, mountain; mount, car; auto; automobile; machine; motorcar, water, sea, field, fence; fencing, rock; stone, signboard; sign, counter, sand, skyscraper, path, runway, river, bridge; span, blind; screen, hill, palm; palm; tree, kitchen; island, boat, bar, bus; autobus; coach; charabanc; double-decker; jitney; motorbus; motorcoach; omnibus; passenger; vehicle, light; light; source, truck; motortruck, tower, awning; sunshade; sunblind, streetlight; street; lamp, booth; cubicle; stall; kiosk, airplane; aeroplane; plane, dirt; track, pole, land; ground; soil, van, ship, fountain, waterfall; falls, tent; collapsible; shelter, minibike; motorbike, tank; storage; tank, lake, hood; exhaust; hood, traffic; light; traffic; signal; stoplight, pier; wharf; wharfage; dock, crt; screen, flag

'background':0

Appendix C

This appendix presents the results of Taskonomy approach and also give a general overview of the role of the Taskonomy results for the pipeline.

Firstly we will discuss surface normals. The rose of the surface normals is to give a better understanding of the physics of the environment presented in the image. Having the surface normals for both query and database images will be able to use that data for both pose estimation and pose verification parts of the pipeline. The idea behind the application of the surface normals is similar to the semantic masks. We will calculate the surface normals of the reprojected database images and compare the results with the surface normals of the query image.

Besides the surface normals, Taskonomy has more than 20 different tasks. We have run all of them on a small sample of the InLoc dataset in order to find out whether there is any task that could be useful for the pipeline. Among all of them, the most promising are the following: edges(2D, 3D), room layout, vanishing points.

Bibliography

Abdulla, Waleed (2017). “Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow”. In: *GitHub repository*.

— (2018). “Splash of Color: Instance Segmentation with Mask R-CNN and TensorFlow”. In: URL: https://engineering.matterport.com/splash-of-color-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46.

Arandjelovic, Relja et al. (2015). “NetVLAD: CNN architecture for weakly supervised place recognition”. In: *CoRR* abs/1511.07247. arXiv: 1511.07247. URL: http://arxiv.org/abs/1511.07247.

Bansal, Aayush, Bryan C. Russell, and Abhinav Gupta (2016). “Marr Revisited: 2D-3D Alignment via Surface Normal Prediction”. In: *CoRR* abs/1604.01347. arXiv: 1604.01347. URL: http://arxiv.org/abs/1604.01347.

Chang, Angel et al. (2017). “Matterport3D: Learning from RGB-D Data in Indoor Environments”. In: *International Conference on 3D Vision (3DV)*.

Cordts, Marius et al. (2016). “The Cityscapes Dataset for Semantic Urban Scene Understanding”. In:

Girshick, Ross B. (2015). “Fast R-CNN”. In: *CoRR* abs/1504.08083. arXiv: 1504.08083. URL: http://arxiv.org/abs/1504.08083.

Harley, Adam W., Konstantinos G. Derpanis, and Iasonas Kokkinos (2017). “Segmentation-Aware Convolutional Networks Using Local Attention Masks”. In: *CoRR* abs/1708.04607. arXiv: 1708.04607. URL: http://arxiv.org/abs/1708.04607.

He, Kaiming et al. (2015). “Deep Residual Learning for Image Recognition”. In: *CoRR* abs/1512.03385. arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

He, Kaiming et al. (2017). “Mask R-CNN”. In: *CoRR* abs/1703.06870. arXiv: 1703.06870. URL: http://arxiv.org/abs/1703.06870.

Lin, Tsung-Yi et al. (2014). “Microsoft COCO: Common Objects in Context”. In: *CoRR* abs/1405.0312. arXiv: 1405.0312. URL: http://arxiv.org/abs/1405.0312.

Lin, Tsung-Yi et al. (2016). “Feature Pyramid Networks for Object Detection”. In: *CoRR* abs/1612.03144. arXiv: 1612.03144. URL: http://arxiv.org/abs/1612.03144.

Ren, Shaoqing et al. (2015). “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *CoRR* abs/1506.01497. arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

Song, Shuran et al. (2017). “Semantic Scene Completion from a Single Depth Image”. In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Taira, Hajime et al. (2018b). “InLoc: Indoor Visual Localization with Dense Matching and View Synthesis”. In:

Taira, Hajime et al. (2018a). “InLoc: Indoor Visual Localization with Dense Matching and View Synthesis”. In: *CoRR* abs/1803.10368. arXiv: 1803.10368. URL: http://arxiv.org/abs/1803.10368.

Wijmans, Erik and Yasutaka Furukawa (2017). “Exploiting 2D Floorplan for Building-scale Panorama RGBD Alignment”. In: URL: http://cvpr17.wijmans.xyz/CVPR2017-0111.pdf.

Wu, Yi et al. (2018). "Building generalizable agents with a realistic and rich 3D environment". In: *arXiv preprint arXiv:1801.02209*.

Zamir, Amir R. et al. (2018). "Taskonomy: Disentangling Task Transfer Learning". In:

Zhang, Yinda et al. (2016). "Physically-Based Rendering for Indoor Scene Understanding Using Convolutional Neural Networks". In: *CoRR* abs/1612.07429. arXiv: 1612.07429. URL: http://arxiv.org/abs/1612.07429.

Zhao, Hengshuang et al. (2017). "Pyramid Scene Parsing Network". In: *CVPR*.

Zhou, Bolei et al. (2017). "Scene Parsing through ADE20K Dataset". In:

Zhou, Bolei et al. (2018). "Semantic understanding of scenes through the ade20k dataset". In: *International Journal on Computer Vision*.