ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Segmentation of Geometrical Shapes in Aerial Images

Master's Thesis

Zuoyue Li

April 2018

Supervisors: Prof. Dr. T. Hofmann, Dr. A. Lucchi, Dr. J. D. Wegner
Department of Computer Science, ETH Zürich

Abstract

This example thesis briefly shows the main features of our thesis style, and how to use it for your purposes.

Acknowledgment

Thanks Mum!

Contents

Chapter 1

Introduction

Dummy text.

1.1 Background

Dummy text.

1.1.1 Aerial Image

Dummy text.

1.1.2 Image Segmentation

Dummy text.

Example Subsubsection

Dummy text.

Example Paragraph Dummy text.

Example Subparagraph Dummy text.

1.1.3 Geometrical Shape

Dummy text.

1.2 Problem Definition

Dummy text.

1.3 Focus of This Work

Dummy text.

1.4 Thesis Organization

Dummy text.

Chapter 2

Related Work

In this chapter, the architectures of PolygonRNN and Faster/Mask R-CNN are presented in detail (see section 3.1 and section 3.2 respectively). Considering our problem, we combine PolygonRNN and FPN part of Mask R-CNN together and come up with a new model, which is called R-PolygonRNN (Region-based PolygonRNN, see section 3.3). In theory, the proposed model should find out the bounding boxes of buildings within an aerial image and give geometrical shape for each building.

2.1 Previous Theses

Dummy text.

2.2 Recent Models

Dummy text.

2.2.1 PolygonRNN

PolygonRNN XXXannotate An RNN is a powerful representation of time-series data, as it carries more complex information about the history by employing linear and non-linear functions. In our case, we hope the RNN to capture the shape of the object and thus make coherent predictions even in ambiguous cases such as for example shadows and saturation. RNNRNN

(right), we feed in an image representation using a modified VGG architecture. Our RNN is a two-layer convolutional LSTM with skip-connection from one and two time steps ago. At the output at each time step, we predict the spatial location of the new vertex of the polygon. At each time step, RNN outputs a vertex with the highest probability.

2.2.2 Mask R-CNN

Dummy text.

2.3 Motivation

Dummy text.

Chapter 3

Model Architecture

In this chapter, the architectures of several models are presented. Section 3.1 gives a comprehensive explanation to the structure of PolygonRNN, while section 3.2 looks into the FPN part of Mask R-CNN. Considering our problem, we combine PolygonRNN and FPN part of Mask R-CNN together and come up with a new model, which is called R-PolygonRNN (Region-based PolygonRNN, see section 3.3). In theory, the proposed model can find out the bounding boxes of buildings within an aerial image and give geometrical shape for each building.

3.1 PolygonRNN

PolygonRNN is the core model for finding geometrical shapes in this project. Figure 3.1 shows the simplified structure of PolygonRNN. The CNN part (see subsection 3.1.1) can capture image features through multilayer convolutions and max pooling, which is then fed into the RNN part (see subsection 3.1.2) to sequentially find out the polygon vertices.

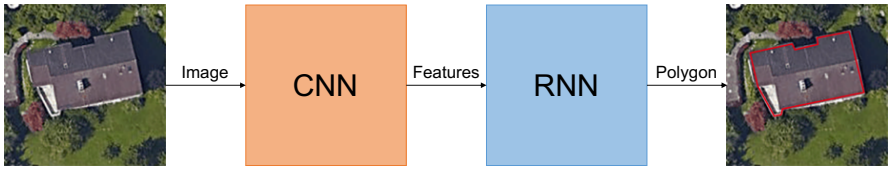


Figure 3.1: Simplified structure of PolygonRNN.

3.1.1 CNN Part

As mentioned in subsection 2.2.1, the CNN part of PolygonRNN uses VGG-16. Actually, VGG-16 is a form of VGGNet, which is proposed by the Visual Geometry Group of Oxford University. VGGNet is very structured and fo-

cusing on deepening the neural network without a large number of parameters. It generally believes that deeper networks have stronger expressive capabilities than shallow networks, and can accomplish more complex tasks. It also proven in practice that VGGNet has made great progress in performance compared to its previous network architecture (e.g. AlexNet).

The '16' in VGG-16 means that it is a VGGNet with 16 layers containing parameters (13 convolutional layers and 3 fully connected layers). It has around 138 million parameters in total. Figure 3.2 shows its detailed network structure. From the figure we can see that VGG-16 continuously does convolution with 3×3 small kernels and makes 2×2 max pooling. As the network deepens, the width and height of the image are reduced by half after each max pooling, and the number of channels is also doubly increasing after some convolution.

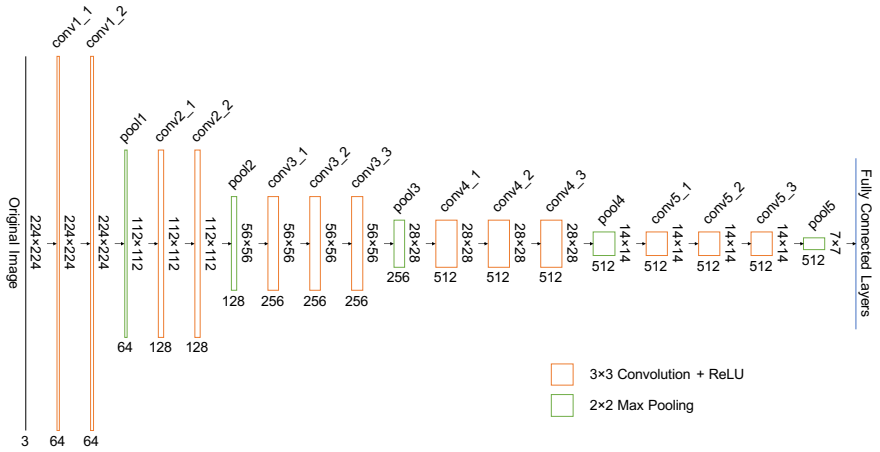


Figure 3.2: VGG-16 architecture. Only convolutional layers and max pooling layers are presented.

VGG-16 was mostly used for image classification before, so after the convolutional layers and max pooling layers there are fully connected layers and softmax layer for the class labels. However, these two kinds of layers are not required for VGG-16 used in PolygonRNN, because the CNN here is working as a feature extractor and mask predictor. Layer `pool5` is omitted as well because of the too low resolution.

Feature Extraction The modified VGG-16 provides RNN with useful features, which are taken from different convolutional and max pooling layers. Specifically, the features are extracted from layer `pool2`, `pool3`, `conv4_3` and `conv5_3`. Note that since the resolution of final features is fixed, when taking features from layer `pool2` and `conv5_3`, it requires another max pooling and upsampling respectively. All of these can be seen in figure 3.3.

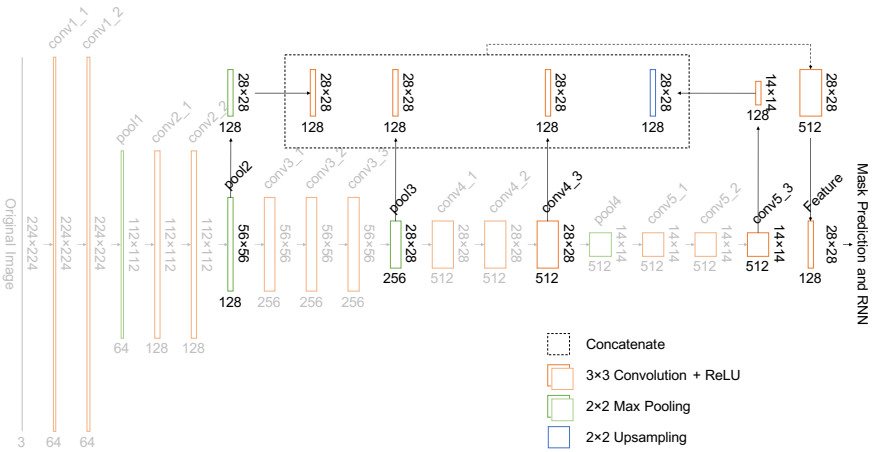


Figure 3.3: Modified VGG-16 architecture in PolygonRNN. In this figure, the highlight layers are used to extract features.

Mask Prediction Another function of the CNN part is to predict masks of boundary and vertices in a low resolution (one eighth of the original). Figure 3.4 shows the mask prediction phase. Different from the ReLU function used in the former convolutional layers, the activation function used here is the sigmoid function. Each entry of the boundary or the vertices mask indicates the probability that the pixel is located in the boundary or is a vertex respectively. The features and two masks are then sent into RNN together.

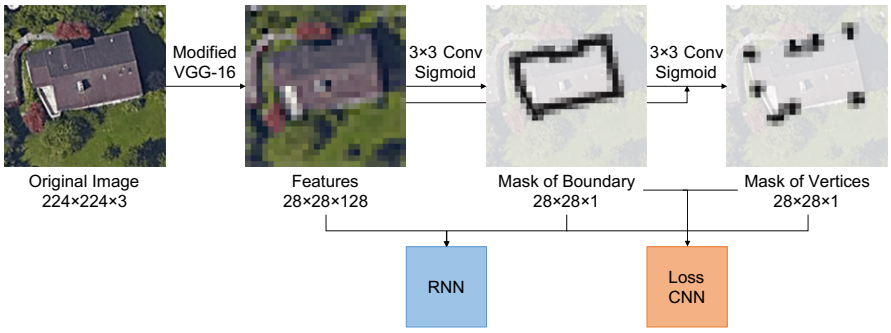


Figure 3.4: Mask prediction of VGG-16. Note that the mask of vertices is obtained by the convolution on the concatenation of the features and the mask of boundary.

3.1.2 RNN Part

The model used for predicting polygon vertices is RNN. We have mentioned in subsection 2.2.1 that RNN is very powerful when data is related to time series, and in our case, we regards the polygon as a series of vertices.

We know that given two vertices on a polygon in an order (either clockwise or anticlockwise), the third vertex after the two points can be uniquely determined. What RNN here can do is to predict the probability distribution of the next vertex's position in a low resolution when given the history information about the two vertices before, as well as the image features and the position of the starting vertex. This process can be formulated as follows.

$$p(v_t|v_{t-1}, v_{t-2}) = F(v_{t-1}, v_{t-2}, v_0, p), \forall t \in \{2, 3, 4, ..., T\}, \tag{3.1}$$

where p denotes the extracted features by CNN (including the masks of boundary and vertices), $F(\cdot)$ denotes a function for computing the conditional probability, T denotes the number of vertices of the polygon, v_t denotes the vertex position at t-th prediction. Specifically, the vertex prediction problem can be formulated as a classification problem. The position of v_t can be then quantized to the resolution of output grid, and we can thus use one-hot encoding for v_t's representation.

End Signal Note that the positions for v_t are not only limited to the general output grid, the end signal for the closure detection of the polygon is also embedded in v_t, just like the 'end of sequence' token `<eos>` or `</s>` in the RNN language model. Thus, the number possible assignments of v_t equals to the resolution of the output grid plus one ($28 \times 28 + 1 = 785$ in our case). In order to correctly predict the end signal, the starting vertex v_0 is required for the conditional probability (equation 3.1) calculation, as it tells the model when to finish the prediction phase. If the current prediction is the same as, or very close to the starting vertex v_0, v_t will be forced to raise the end signal, indicating that the entire polygon is close, and the prediction phase is therefore complete. So generally, the end signal works when $t = T$.

Starting Vertex In equation 3.1, two special cases $p(v_0)$ and $p(v_1|v_0)$ are not included yet. These two cases are different from the general case, and should be considered in addition. In particular, we can directly regard the mask of vertices (for example, the rightmost image in figure 3.4) predicted by the CNN as v_0's unnormalized probability distribution $\tilde{p}(v_0)$, and choose the position with the highest probability for v_0's assignment. As v_0 is known, there are typically two options for v_1, one is the next vertex on its left direction and another on right direction. To tackle this problem, we can simply specify the order of the polygon vertices to be fixed, so that v_1 can be uniquely determined. In our project, the order of polygon is set to be anticlockwise.

ConvLSTM As mentioned in subsection 2.2.1 the model uses ConvLSTM cell as the polygon decoder to sequentially predict vertices. For simplicity, we can regard ConvLSTM as the function $F(\cdot)$ in equation 3.1. In fact, the structure of a ConvLSTM cell is almost the same as that of an ordinary LSTM

cell, except that it employs convolution in a 2D image with multiple channels instead of matrices or vectors multiplication. The introduction of ConvLSTM can significantly reduce the the number of parameters when it is compared with a fully connected RNN. The following equations define the computation process within a ConvLSTM cell, which is also visualized in figure 3.5.

$$\begin{bmatrix} f_t \\ i_t \\ g_t \\ o_t \end{bmatrix} = \begin{bmatrix} W_{hf} \\ W_{hi} \\ W_{hg} \\ W_{ho} \end{bmatrix} * h_{t-1} + \begin{bmatrix} W_{xf} \\ W_{xi} \\ W_{xg} \\ W_{xo} \end{bmatrix} * x_t + \begin{bmatrix} b_f \\ b_i \\ b_g \\ b_o \end{bmatrix} = W_h * h_{t-1} + W_x * x_t + b, \tag{3.2}$$

$$c_t = \sigma(f_t) \circ c_{t-1} + \sigma(i_t) \circ \tanh(g_t), \tag{3.3}$$

$$h_t = \sigma(o_t) \circ \tanh(c_t), \tag{3.4}$$

where x_t, h_t, c_t denote the input, the hidden state (or cell output), and the cell state of the cell at time step t respectively, i_t, o_t, f_t denote the states of input, output, and forget gate at time step t respectively, g_t denotes an intermediate variable, $\sigma(\cdot)$, $*$, $\circ$ denote the sigmoid function, convolution, and Hadamard (element-wise) product respectively, W_x and W_h denotes two convolution kernels for x_t and h_t respectively, W_{xf}, W_{xi}, W_{xg}, W_{xo} denote the four components of W_x and W_{hf}, W_{hi}, W_{hg}, W_{ho} denote the four components of W_h.

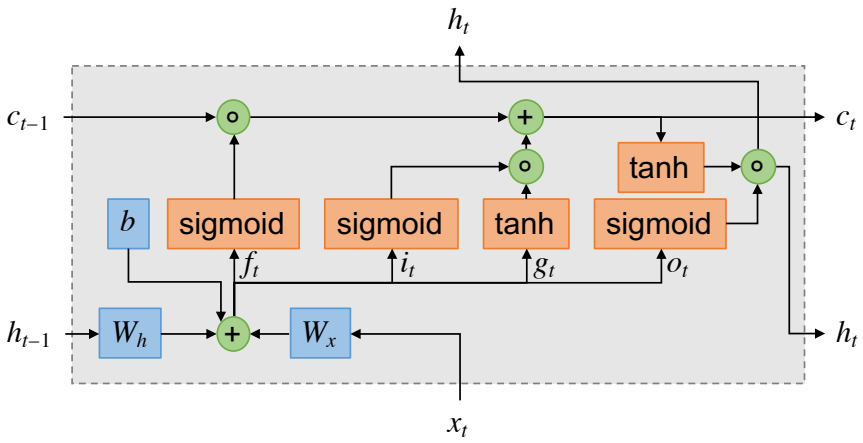


Figure 3.5: Visualization for LSTM cell.

Multilayer RNN PolygonRNN uses multilayer RNN with ConvLSTM cells. The connection between two neighboring layers can be described as follows, which means that the cell input of this layer comes from the cell output of the previous layer.

$$x_t^{(k)} = h_t^{(k-1)}, \forall k \in \{2, 3, ..., n\}, \tag{3.5}$$

where n is the number of RNN layers or ConvLSTM cells, k in the superscript denotes the k-th layer. Recall that the initial cell input (i.e. the cell input of the first layer) at time step t consists of the spatial information p from CNN, the two previous vertices v_{t-1} and v_{t-2}, and starting vertex v_0, here it is formulated as follows.

$$x_t = x_t^{(1)} = v_{t-1} \oplus v_{t-2} \oplus v_0 \oplus p, \tag{3.6}$$

where $\oplus$ denotes the concatenation operation in the dimension of image channel. Thus, the equation 3.1 can be written as follows.

$$y_t = p(v_t | v_{t-1}, v_{t-2}) = \text{softmax}(W_y \, \text{vec}(h_t^{(n)})), \tag{3.7}$$

where W_y denotes the weights of the final fully connected layer. Figure 3.6 shows the entire structure of the RNN part and highlights the process at time step t, using the same notation as the equations 3.5, 3.6 and 3.7.

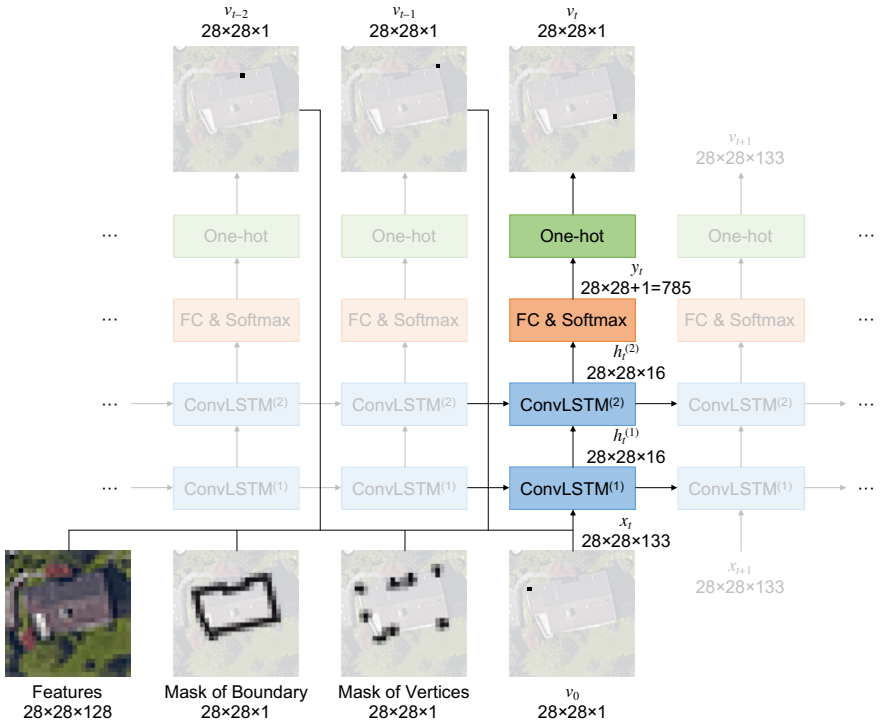


Figure 3.6: Visualization for the time step of the RNN decoder. In this figure, the outputs of the end signal are omitted, and the configuration of the ConvLSTM cell is the same as what is used in the original PolygonRNN paper. Specifically, the paper sets the number of RNN layers to 2 and uses ConvLSTM cells with 3×3 kernel size and 16 channels.

3.1.3 Training

the training of PolygonRNN

Loss of CNN Part Cross-entropy loss

The

The two masks are also used to compute the loss as well when given the ground truth. Here the weighted log loss is adopted.

Loss of RNN Part

3.2 Faster/Mask R-CNN

Dummy text.

3.2.1 Region Proposal Network

Dummy text.

3.2.2 Feature Pyramid Network

Dummy text.

3.3 R-PolygonRNN

Dummy text.

3.3.1 Two-step Model

Dummy text.

3.3.2 Hybrid Model

Dummy text.

Chapter 4

Experiments and Results

In this chapter, the entire experimental process is presented, from the acquisition of the ground truth dataset (see section 4.1), to the implementation, training and prediction phases of the R-PolygonRNN (see section 4.2), and finally to the results evaluation and analysis (see section 4.3).

4.1 Ground Truth

Our ground truth dataset consists of satellite images and buildings' coordinates, which are used as inputs and labels respectively when training. In this project, all of the satellite images are collected from Google Static Maps API and all of the latitude and longitude coordinates of the polygon vertices of buildings are collected from OpenStreetMap. For details of the two APIs mentioned above, please refer to subsections 4.1.1 and 4.1.2.

As mentioned in section 3.3, the training phase of our model R-PolygonRNN requires two different kinds of ground truth dataset, areas with multiple bounding boxes for FPN and buildings with geometrical shapes for PolygonRNN, all of which are illustrated in subsection 4.1.3.

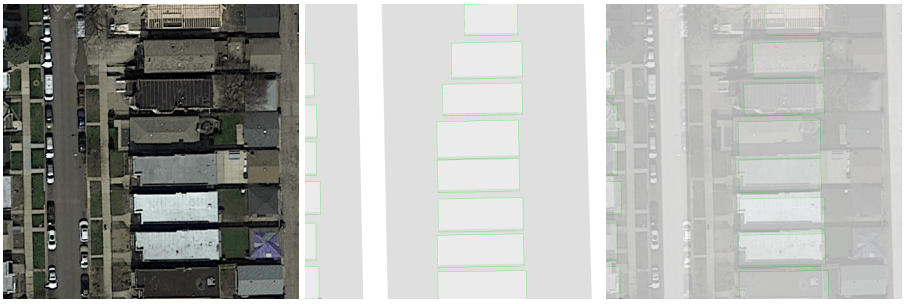
Since the whole dataset is collected from two different sources, the problem of inconsistency may exist. Subsection 4.1.4 describes details of the problems in the ground truth dataset, and subsection 4.1.5 proposes a solution to adjust the shift between buildings' images and polygons.

4.1.1 Google Static Maps API

Google Static Maps API[1] provides an interface that implements maps as high-resolution images. Users can download customized map based on URL with different parameters, which is sent through a standard HTTPS request.

[1] `https://developers.google.com/maps/documentation/static-maps/`

The parameters in URL includes the map type (satellite, roadmap, etc.), latitude and longitude coordinates of the image center, the resolution of the image, the zoom level, and the scale. For the usage of the Google Static Maps API, please refer to section A.1 in appendices. Figure 4.1(a) and 4.1(b) shows two types of images can be obtained by Google Static Maps API.



(a) satellite image (b) roadmap image (c) hybrid image

Figure 4.1: Example images downloaded through Google Static Maps API. (a) and (b) are obtained directly by the API, with 41.9399708 degrees north latitude, 87.7380649 degrees west longitude of the center (located in Chicago), both width and height 600 pixels, zoom level 20 and scale 1, but different in map types. (c) is obtained by overlaying (a) with (b), with 75% opacity. It can be clearly seen that the two images cannot match very well.

We would have liked to obtain buildings' coordinates from Google Static Maps API as well, but we find that the API does not provide such information directly. Instead, it gives the styled roadmap images like figure 4.1(b), where querying coordinates requires further corner detection but the boundaries may be still inaccurate (see figure 4.1(c) for example). Thus, this thesis project, only satellite images from Google Static Maps API are used.

4.1.2 OpenStreetMap

OpenStreetMap[2] provides an interface that implements a map as a XML format `.osm` file. The file contains all the information which can be used to recover the map. When the map's bounding box is given, users can retrieve the latitude and longitude coordinates of the buildings' vertices and roads' central lines within the map. For the usage of OpenStreetMap and the format of the `.osm` file, please refer to section A.2 in appendices.

The coordinates obtained by the API is the original latitude and longitude coordinates of the buildings' vertices, which cannot be directly used for training. We need to convert them to relative pixel coordinates with regards to an given satellite image.

[2] `https://www.openstreetmap.org/`

As a matter of fact, every fixed point on the earth can correspond to a unique pixel on the map at a specific zoom level. This projection process can be regarded as a function. Since we have already known the relative pixel coordinates of the image center (half width and half height), the projection for an arbitrary point with given latitude and longitude coordinates can be therefore computed. This process can be described as following equations.

$$(c_x, c_y) = (\frac{w}{2}, \frac{h}{2}) = f(c_{lon}, c_{lat}, z) + (\Delta_x, \Delta_y), \tag{4.1}$$

$$(p_x, p_y) = f(p_{lon}, p_{lat}, z) + (\Delta_x, \Delta_y), \tag{4.2}$$

where f is the function that projects latitude and longitude coordinates to global pixel coordinates (see A.3 in appendices for more details), c is the image center, p is an arbitrary point, the subscripts x, y, lon, lat mean the relative pixel and longitude and latitude coordinates respectively, z is the zoom level, Δ is the translation constant from the global to the relative pixel coordinate system. Thus, we have

$$(p_x, p_y) = f(p_{lon}, p_{lat}, z) - f(c_{lon}, c_{lat}, z) + (\frac{w}{2}, \frac{h}{2}), \tag{4.3}$$

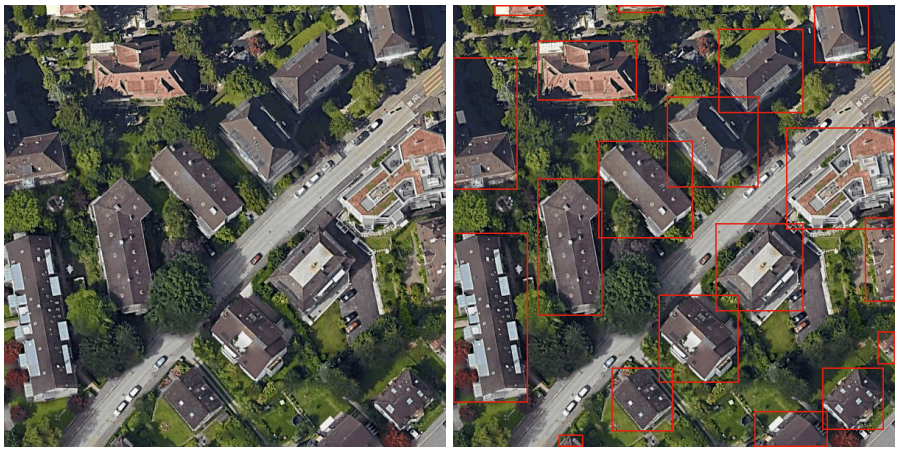
for any arbitrary point p.

4.1.3 Areas and Buildings

As mentioned in section 3.3, R-PolygonRNN is formed by FPN and PolygonRNN. However, the training phase of these two models requires two different kinds of ground truth dataset.

Areas for FPN In subsection 3.2.2 we have shown that FPN aims at finding rectangular regions of interests. Thus, training FPN generally requires a relatively large image with several objects in it. When considering our problem, an example of the ground truth can be an area of a city with several bounding boxes. Each box contains a single building, regardless of its tight polygon outline. Figure 4.2 gives an example area for training FPN and its visualized ground truth label.

Buildings for PolygonRNN In section 3.1 we have mentioned that PolygonRNN can only deal with images with single object, meaning that the bounding box of the object should be first given. Thus, training PolygonRNN generally requires a relatively small image with single object, as well as the information of its polygon outline. When it comes to our problem, an example of the ground truth can be a building with some padding and the pixel coordinates of building's vertices. Figure 4.3 gives example buildings for training PolygonRNN and its visualized ground truth label.



(a) example area (b) example area with bounding boxes

Figure 4.2: An example area in Zurich for FPN training. (a) is the original satellite image obtained by Google Static Maps API. (b) is (a) covered by multiple bounding boxes of buildings, which is visualized based on the ground truth.

(a) example buildings

(b) example buildings with polygon outlines

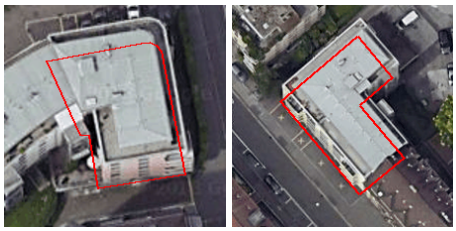
Figure 4.3: Example buildings in Zurich for PolygonRNN training. (a) is the original satellite images obtained by Google Static Maps API. (b) is (a) covered by the corresponding polygons' edges, which are visualized based on the ground truth.

4.1.4 Problems

Recall that the satellite images and polygons' coordinates are collected from two different sources, the problem of inconsistency may exist. That is to say, the polygon and the actual building we see in the image may not match well in some cases. This inconsistency is mainly reflected in the following aspects.

(a) general shift

(b) polygon size and shape



(c) angle issue

(d) adjacent buildings

Figure 4.4: Problems existed in the ground truth dataset.

General Shift Figure 4.4(a) gives two typical shift examples in the dataset. This shift is generally within the range of 30 pixels, either up and down, or left and right.

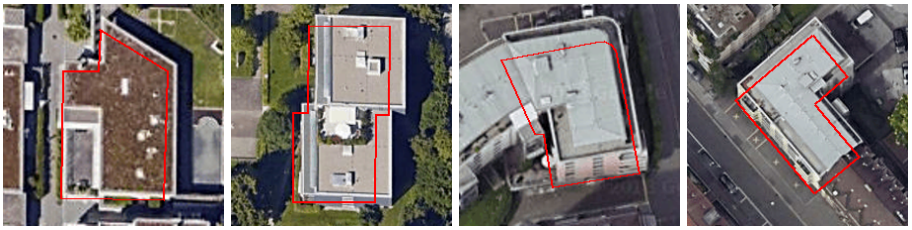
Polygon Size and Shape Sometimes the polygon size and shape is different from the actual building size as well. This is typically because OpenStreetMap measures the bottom of the building and the aerial image is taken from the top. The roof of the house we see from the top can be different from the floor area. Figure 4.4(b) shows a building with a smaller polygon and a building with a too rough outline.

Angle Issue Some tall buildings may suffer from the angle issue. This is because the satellite cannot always be vertically right above the building, so the side of these very high buildings will be photographed. Figure 4.4(c) gives two examples.

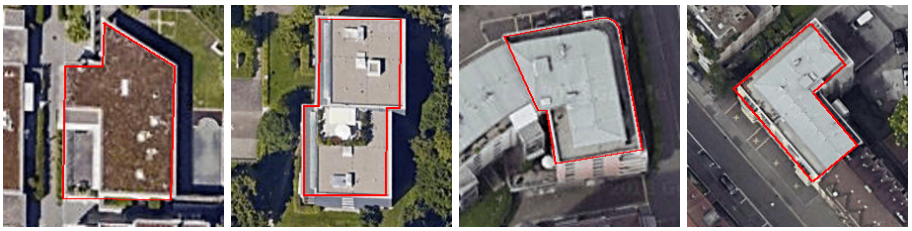
Adjacent Buildings Some adjacent buildings will be very close together, and some even share a common roof. Thus, from the satellite image, it is generally difficult to distinguish the boundaries between them. However, these buildings are separated in the dataset of OpenStreetMap. It means that where there are originally edges between buildings, it looks like there is no edge in the image. This kind of inaccuracy would have negative effect on the training phase. Figure 4.4(d) and gives two examples.

4.1.5 Adjustments

Problems such as too rough polygon shape, angle issue and adjacent buildings are unsolvable unless the data sources can correct themselves. What we can improve is to tackle the general shift and the polygon size problem and make the image-polygon pairs more matching. By observation, we can get to some conclusions for a matching image-polygon pair.



(a) before adjustment



(b) after adjustment

Figure 4.5: Adjustment examples. (a) shows the polygons before adjustment, while (b) shows the polygons after adjustment.

Color Variance The color variance of image pixels covered by polygon generally reaches the local minimum, because the roof of a building usually has only one color or several similar colors.

Edges The polygon edges generally lie within the output of the edge detection (e.g. Sobel, Canny) performed on the image.

Corners The polygon vertices can generally match the output of the corner detection (e.g. Harris, SIFT) performed on the image.

Based on these conclusions, we propose a brute force shift adjustment method. Briefly, we resize and translate the initial polygon within a certain range, and traverse all the cases to see where the minimum color variance, maximum edge and corner coverage are reached. Figure 4.5 illustrates some examples which compare the polygons before and after the adjustment. Results show that this algorithm can well address the shift problem to some extent.

4.2 Implementation Details

4.2.1 Configuration

number of layers LSTM polygon anticlockwise remove point in a line

4.2.2 Training

Furthermore, buildings around the edges of areas are typically ignored

more likely to be incomplete. In this case, the polygon would be clipped by the edges, resulting in new vertices, which can be very hard to compute.

(*figureplaceholder*)

Although buildings can extract from areas

Therefore, in this project, two different kinds of ground truth dataset are collected separately. batchsizeBuildingareaPolygonRNNbatchsize

4.2.3 Prediction

Beam search

4.3 Experiment Results

Dummy text.

4.3.1 Single Building Segmentation

Dummy text.

4.3.2 Buildings Localization

Dummy text.

4.3.3 R-PolygonRNN

Beam search

Example Subsubsection

Dummy text.

Example Paragraph Dummy text.

Example Subparagraph Dummy text.

Chapter 5

Problems and Future Work

Dummy text.

5.1 Problems

Dummy text.

5.1.1 Problem 1

Dummy text.

5.1.2 Problem 2

Dummy text.

Example Subsubsection

Dummy text.

Example Paragraph Dummy text.

Example Subparagraph Dummy text.

5.2 Future Work

Dummy text.

5.2.1 Ground Truth

Dummy text.

5.2.2 Training Phase

Dummy text.

5.2.3 Model Generalization

Dummy text.

Chapter 6

Introduction

This is version `v1.4` of the template.

We assume that you found this template on our institute's website, so we do not repeat everything stated there. Consult the website again for pointers to further reading about LaTeX. This chapter only gives a brief overview of the files you are looking at.

6.1 Features

The rest of this document shows off a few features of the template files. Look at the source code to see which macros we used!

The template is divided into TeX files as follows:

1. `thesis.tex` is the main file.
2. `extrapackages.tex` holds extra package includes.
3. `layoutsetup.tex` defines the style used in this document.
4. `theoremsetup.tex` declares the theorem-like environments.
5. `macrosetup.tex` defines extra macros that you may find useful.
6. `introduction.tex` contains this text.
7. `sections.tex` is a quick demo of each sectioning level available.
8. `refs.bib` is an example bibliography file. You can use BibTeX to quote references. For example, read [**?**] if you can get a hold of it.

6.1.1 Extra package includes

The file `extrapackages.tex` lists some packages that usually come in handy. Simply have a look at the source code. We have added the following comments based on our experiences:

REC This package is recommended.

OPT This package is optional. It usually solves a specific problem in a clever way.

ADV This package is for the advanced user, but solves a problem frequent enough that we mention it. Consult the package's documentation.

As a small example, here is a reference to the Section *Features* typeset with the recommended *varioref* package:

> See Section 6.1 on the preceding page.

6.1.2 Layout setup

This defines the overall look of the document – for example, it changes the chapter and section heading appearance. We consider this a 'do not touch' area. Take a look at the excellent *Memoir* documentation before changing it.

In fact, take a look at the excellent *Memoir* documentation, full stop.

6.1.3 Theorem setup

This file defines a bunch of theorem-like environments.

Theorem 6.1 *An example theorem.*

Proof Proof text goes here. □

Note that the q.e.d. symbol moves to the correct place automatically if you end the proof with an `enumerate` or `displaymath`. You do not need to use `\qedhere` as with *amsthm*.

Theorem 6.2 (Some Famous Guy) *Another example theorem.*

Proof This proof

1. ends in an enumerate. □

Proposition 6.3 *Note that all theorem-like environments are by default numbered on the same counter.*

Proof This proof ends in a display like so:

$$f(x) = x^2. \qquad \square$$

6.1.4 Macro setup

For now the macro setup only shows how to define some basic macros, and how to use a neat feature of the *mathtools* package:

$$|a|, \quad \left|\frac{a}{b}\right|, \quad |\frac{a}{b}|.$$

Chapter 7

Writing scientific texts in English

This chapter was originally a separate document written by Reto Sphel. It is reprinted here so that the template can serve as a quick guide to thesis writing, and to provide some more example material to give you a feeling for good typesetting.

7.1 Basic writing rules

The following rules need little further explanation; they are best understood by looking at the example in the booklet by Knuth et al., 2–3.

Rule 7.1 Write texts, not chains of formulas.

More specifically, write full sentences that are logically interconnected by phrases like 'Therefore', 'However', 'On the other hand', etc. where appropriate.

Rule 7.2 Displayed formulas should be embedded in your text and punctuated with it.

In other words, your writing should not be divided into 'text parts' and 'formula parts'; instead the formulas should be tied together by your prose such that there is a natural flow to your writing.

7.2 Being nice to the reader

Try to write your text in such a way that a reader enjoys reading it. That's of course a lofty goal, but nevertheless one you should aspire to!

Rule 7.3 Be nice to the reader.

Give some intuition or easy example for definitions and theorems which might be hard to digest. Remind the reader of notations you introduced

many pages ago – chances are he has forgotten them. Illustrate your writing with diagrams and pictures where this helps the reader. Etc.

Rule 7.4 Organize your writing.

Think carefully about how you subdivide your thesis into chapters, sections, and possibly subsections. Give overviews at the beginning of your thesis and of each chapter, so the reader knows what to expect. In proofs, outline the main ideas before going into technical details. Give the reader the opportunity to 'catch up with you' by summing up your findings periodically.

Useful phrases: 'So far we have shown that ...', 'It remains to show that ...', 'Recall that we want to prove inequality (7), as this will allow us to deduce that ...', 'Thus we can conclude that .... Next, we would like to find out whether ...', etc.

Rule 7.5 Don't say the same thing twice without telling the reader that you are saying it twice.

Repetition of key ideas is important and helpful. However, if you present the same idea, definition or observation twice (in the same or different words) without telling the reader, he will be looking for something new where there is nothing new.

Useful phrases: 'Recall that [we have seen in Chapter 5 that] ...', 'As argued before / in the proof of Lemma 3, ...', 'As mentioned in the introduction, ...', 'In other words, ...', etc.

Rule 7.6 Don't make statements that you will justify later without telling the reader that you will justify them later.

This rule also applies when the justification is coming right in the next sentence! The reasoning should be clear: if you violate it, the reader will lose valuable time trying to figure out on his own what you were going to explain to him anyway.

Useful phrases: 'Next we argue that ...', 'As we shall see, ...', 'We will see in the next section that ..., etc.

7.3 A few important grammar rules

Rule 7.7 There is (almost) *never* a comma before 'that'.

It's really that simple. Examples:

> We assume that ...
> *Wir nehmen an, dass ...*

It follows that ...
Daraus folgt, dass ...

'thrice' is a word that is seldom used.
'thrice' ist ein Wort, das selten verwendet wird.

Exceptions to this rule are rare and usually pretty obvious. For example, you may end up with a comma before 'that' because 'i.e.' is spelled out as 'that is':

For $p(n) = \log n/n$ we have ... However, if we choose p a little bit higher, that is $p(n) = (1+\varepsilon)\log n/n$ for some $\varepsilon > 0$, we obtain that...

Or you may get a comma before 'that' because there is some additional information inserted in the middle of your sentence:

Thus we found a number, namely n_0, that satisfies equation (13).

If the additional information is left out, the sentence has no comma:

Thus we found a number that satisfies equation (13).

(For 'that' as a relative pronoun, see also Rules 7.9 and 7.10 below.)

Rule 7.8 There is usually no comma before 'if'.

Example:

A graph is not 3-colorable if it contains a 4-clique.
Ein Graph ist nicht 3*-frbbar, wenn er eine* 4*-Clique enthlt.*

However, if the 'if' clause comes first, it is usually separated from the main clause by a comma:

If a graph contains a 4-clique, it is not 3-colorable .
Wenn ein Graph eine 4*-Clique enthlt, ist er nicht* 3*-frbbar.*

There are more exceptions to these rules than to Rule 7.7, which is why we are not discussing them here. Just keep in mind: don't put a comma before 'if' without good reason.

Rule 7.9 Non-defining relative clauses have commas.

Rule 7.10 Defining relative clauses have no commas.

In English, it is very important to distinguish between two types of relative clauses: defining and non-defining ones. This is a distinction you absolutely need to understand to write scientific texts, because mistakes in this area actually distort the meaning of your text!

It's probably easier to explain first what a *non-defining* relative clause is. A non-defining relative clauses simply gives additional information *that could also be left out* (or given in a separate sentence). For example, the sentence

> The WEIRDSORT algorithm, which was found by the famous mathematician John Doe, is theoretically best possible but difficult to implement in practice.

would be fully understandable if the relative clause were left out completely. It could also be rephrased as two separate sentences:

> The WEIRDSORT algorithm is theoretically best possible but difficult to implement in practice. [By the way,] WEIRDSORT was found by the famous mathematician John Doe.

This is what a non-defining relative clause is. *Non-defining relative clauses are always written with commas.* As a corollary we obtain that you cannot use 'that' in non-defining relative clauses (see Rule 7.7!). It would be wrong to write

> ~~The WEIRDSORT algorithm, that was found by the famous mathematician John Doe, is theoretically best possible but difficult to implement in practice.~~

A special case that warrants its own example is when 'which' is referring to the entire preceding sentence:

> Thus inequality (7) is true, which implies that the Riemann hypothesis holds.

As before, this is a non-defining relative sentence (it could be left out) and therefore needs a comma.

So let's discuss *defining* relative clauses next. A defining relative clause tells the reader *which specific item the main clause is talking about*. Leaving it out either changes the meaning of the sentence or renders it incomprehensible altogether. Consider the following example:

> The WEIRDSORT algorithm is difficult to implement in practice. In contrast, the algorithm that we suggest is very simple.

Here the relative clause 'that we suggest' cannot be left out – the remaining sentence would make no sense since the reader would not know which algorithm it is talking about. This is what a defining relative clause is. *Defining relative clauses are never written with commas.* Usually, you can use both 'that' and 'which' in defining relative clauses, although in many cases 'that' sounds better.

As a final example, consider the following sentence:

> For the elements in $\mathcal{B}$ which satisfy property (A), we know that equation (37) holds.

Table 7.1: Things you (usually) don't say

~~It holds (that) ...~~	We have ...	*Es gilt ...*
('Equation (5) holds.' is fine, though.)		
~~x fulfills property $\mathcal{P}$.~~	x satisfies property $\mathcal{P}$.	*x erfllt Eigenschaft $\mathcal{P}$.*
~~in average~~	on average	*im Durchschnitt*
~~estimation~~	estimate	*Abschtzung*
~~composed number~~	composite number	*zusammengesetzte Zahl*
~~with the help of~~	using	*mit Hilfe von*
~~surely~~	clearly	*sicher, bestimmt*
~~monotonously increasing~~	monotonically incr.	*monoton steigend*
(Actually, in most cases 'increasing' is just fine.)		

This sentence does not make a statement about all elements in $\mathcal{B}$, only about those satisfying property (A). The relative clause is *defining*. (Thus we could also use 'that' in place of 'which'.)

In contrast, if we add a comma the sentence reads

> For the elements in $\mathcal{B}$, which satisfy property (A), we know that equation (37) holds.

Now the relative clause is *non-defining* – it just mentions in passing that all elements in $\mathcal{B}$ satisfy property (A). The main clause states that equation (37) holds for *all* elements in $\mathcal{B}$. See the difference?

7.4 Things you (usually) don't say in English – and what to say instead

Table 7.1 lists some common mistakes and alternatives. The entries should not be taken as gospel – they don't necessarily mean that a given word or formulation is wrong under all circumstances (obviously, this depends a lot on the context). However, in nine out of ten instances the suggested alternative is the better word to use.

Chapter 8

Typography

8.1 Punctuation

Rule 8.1 Use opening (‘) and closing (’) quotation marks correctly.

In LaTeX, the closing quotation mark is typed like a normal apostrophe, while the opening quotation mark is typed using the French *accent grave* on your keyboard (the *accent grave* is the one going down, as in *frre*).

Note that any punctuation that *semantically* follows quoted speech goes inside the quotes in American English, but outside in Britain. Also, Americans use double quotes first. Oppose

> “Using ‘lasers,’ we punch a hole in . . . the Ozone Layer,” Dr. Evil said.

to

> ‘Using “lasers”, we punch a hole in . . . the Ozone Layer’, Dr. Evil said.

Rule 8.2 Use hyphens (-), en-dashes (–) and em-dashes (—) correctly.

A hyphen is only used in words like ‘well-known’, ‘3-colorable’ etc., or to separate words that continue in the next line (which is known as hyphenation). It is entered as a single ASCII hyphen character (-).

To denote ranges of numbers, chapters, etc., use an en-dash (entered as two ASCII hyphens --) with no spaces on either side. For example, using Equations (1)–(3), we see. . .

As the equivalent of the German *Gedankenstrich*, use an en-dash with spaces on both sides – in the title of Section 7.4, it would be wrong to use a hyphen instead of the dash. (Some English authors use the even longer emdash (—)

instead, which is typed as three subsequent hyphens in LaTeX. This emdash is used without spaces around it—like so.)

8.2 Spacing

Rule 8.3 Do not add spacing manually.

You should never use the commands `\\` (except within tabulars and arrays), `\␣` (except to prevent a sentence-ending space after Dr. and such), `\vspace`, `\hspace`, etc. The choices programmed into LaTeX and this style should cover almost all cases. Doing it manually quickly leads to inconsistent spacing, which looks terrible. Note that this list of commands is by no means conclusive.

Rule 8.4 Judiciously insert spacing in maths where it helps.

This directly contradicts Rule 8.3, but in some cases TeX fails to correctly decide how much spacing is required. For example, consider

$$f(a,b) = f(a+b,a-b).$$

In such cases, inserting a thin math space `\,` greatly increases readability:

$$f(a,b) = f(a+b,\, a-b).$$

Along similar lines, there are variations of some symbols with different spacing. For example, Lagrange's Theorem states that $|G| = [G : H]|H|$, but the proof uses a bijection $f\colon aH \to bH$. (Note how the first colon is symmetrically spaced, but the second is not.)

Rule 8.5 Learn when to use `\␣` and `\@`.

Unless you use 'french spacing', the space at the end of a sentence is slightly larger than the normal interword space.

The rule used by TeX is that any space following a period, exclamation mark or question mark is sentence-ending, except for periods preceded by an uppercase letter. Inserting `\` before a space turns it into an interword space, and inserting `\@` before a period makes it sentence-ending. This means you should write

```
Prof.\ Dr.\ A. Steger is a member of CADMO\@.
If you want to write a thesis with her, you
should use this template.
```

which turns into

> Prof. Dr. A. Steger is a member of CADMO. If you want to write a thesis with her, you should use this template.

The effect becomes more dramatic in lines that are stretched slightly during justification:

> Prof. Dr. A. Steger is a member of CADMO. If you

Rule 8.6 Place a non-breaking space (`~`) right before references.

This is actually a slight simplification of the real rule, which should invoke common sense. Place non-breaking spaces where a line break would look 'funny' because it occurs right in the middle of a construction, especially between a reference type (Chapter) and its number.

8.3 Choice of 'fonts'

Professional typography distinguishes many font attributes, such as family, size, shape, and weight. The choice for sectional divisions and layout elements has been made, but you will still occasionally want to switch to something else to get the reader's attention. The most important rule is very simple.

Rule 8.7 When emphasising a short bit of text, use `\emph`.

In particular, *never* use bold text (`\textbf`). Italics (or Roman type if used within italics) avoids distracting the eye with the huge blobs of ink in the middle of the text that bold text so quickly introduces.

Occasionally you will need more notation, for example, a consistent typeface used to identify algorithms.

Rule 8.8 Vary one attribute at a time.

For example, for WEIRDSORT we only changed the shape to small caps. Changing two attributes, say, to bold small caps would be excessive (LaTeX does not even have this particular variation). The same holds for mathematical notation: the reader can easily distinguish g_n, $G(x)$, $\mathcal{G}$ and $\mathtt{G}$.

Rule 8.9 Never underline or uppercase.

No exceptions to this one, unless you are writing your thesis on a typewriter. Manually. Uphill both ways. In a blizzard.

8.4 Displayed equations

Rule 8.10 Insert paragraph breaks *after* displays only where they belong. Never insert paragraph breaks *before* displays.

LaTeX translates sequences of more than one linebreak (i.e., what looks like an empty line in the source code) into a paragraph break in almost all contexts. This also happens before and after displays, where extra spacing is inserted to give a visual indication of the structure. Adding a blank line in these places may look nice in the sources, but compare the resulting display

$$a = b$$

to the following:

$$a = b$$

The first display is surrounded by blank lines, but the second is not. It is bad style to start a paragraph with a display (you should always tell the reader what the display means first), so the rule follows.

Rule 8.11 Never use `eqnarray`.

It is at the root of most ill-spaced multiline displays. The *amsmath* package provides better alternatives, such as the `align` family

$$\begin{aligned} f(x) &= \sin x, \\ g(x) &= \cos x, \end{aligned}$$

and `multline` which copes with excessively long equations:

$$\begin{aligned} &\mathrm{P}\big[X_{t_0} \in (z_0, z_0 + dz_0], \ldots, X_{t_n} \in (z_n, z_n + dz_n]\big] \\ &\qquad = \nu(dz_0) K_{t_1}(z_0, dz_1) K_{t_2 - t_1}(z_1, dz_2) \cdots K_{t_n - t_{n-1}}(z_{n-1}, dz_n). \end{aligned}$$

8.5 Floats

By default this style provides floating environments for tables and figures. The general structure should be as follows:

```
\begin{figure}
  \centering
  % content goes here
  \caption{A short caption}
  \label{some-short-label}
\end{figure}
```

Note that the label must follow the caption, otherwise the label will refer to the surrounding section instead. Also note that figures should be captioned at the bottom, and tables at the top.

The whole point of floats is that they, well, *float* to a place where they fit without interrupting the text body. This is a frequent source of confusion and changes; please leave it as is.

Rule 8.12 Do not restrict float movement to only 'here' (`h`).

If you are still tempted, you should avoid the float altogether and just show the figure or table inline, similar to a displayed equation.

Appendix A

Appendix

A.1 Example URL of Google Static Maps API

```
https://maps.googleapis.com/maps/api/staticmap?maptype=satellite&center=4
&style=feature:all|element:labels|visibility:off&style=feature:landscape.
```

A.2 OpenStreetMap

According to the Google Maps JavaScript API [1],

A.3 Projection

[1] https://developers.google.com/maps/documentation/javascript/coordinates

List of Figures

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.