A Deep Learning Approach to Camera Pose Estimation

Federico Izzo Francesco Bozzo

University of Trento

January 12, 2022

Introduction

With this work we are going to present:

- the camera pose estimation problem;
- the exploration of labeled dataset generation techniques applied on Povo 1 second floor;
- relative and absolute pose estimation deep learning models;
- the post-processing of the model outputs;
- the model deployment using a FastAPI web-server.

Camera Pose Estimation

The **camera pose estimation** problem aims to find the function E that, given an **image** I_c, returns the **position** x_c and **orientation** q_c of the camera.

$$E(I_c) = (x_c, q_c)$$

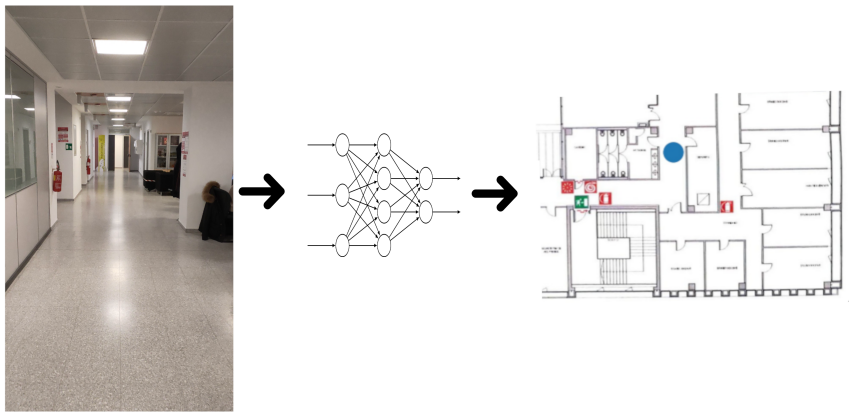

Figure: Example of camera relocalization.

Different Approaches for Dataset Generation

We have considered four different techniques:

- **IMU sensors**: not enough accurate;
- **digital video**: not available in the university environment;
- **motion capture system**: difficult to associate with a recorder video;
- **structure from motion**: our choice (COLMAP).

COLMAP - Structure From Motion

COLMAP is **structure from motion** tool that allows to reconstruct a 3D representation of an environment using images of it. During the process, it also computes camera poses in an arbitrary reference system.

Figure: COLMAP reconstruction of central Rome using 21K photos.

Povo 1 Second Floor Dataset

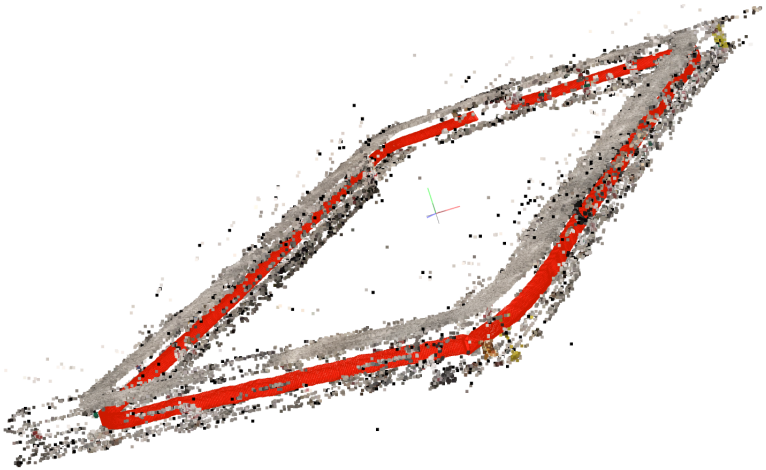

Figure: COLMAP reconstruction of Povo 1 second floor.

After many failed attempts, we managed to create a COLMAP reconstruction of **Povo 1 second floor**. The red line represents the camera trajectory and other points are the extracted features.

Coordinate Reference System Alignment

Some steps are required to align COLMAP's arbitrary **coordinate reference system (CRS)** with respect to the real world:

1. **scale**: align unit of measures;
2. **translate**: align origins;
3. **rotate**: align axes.

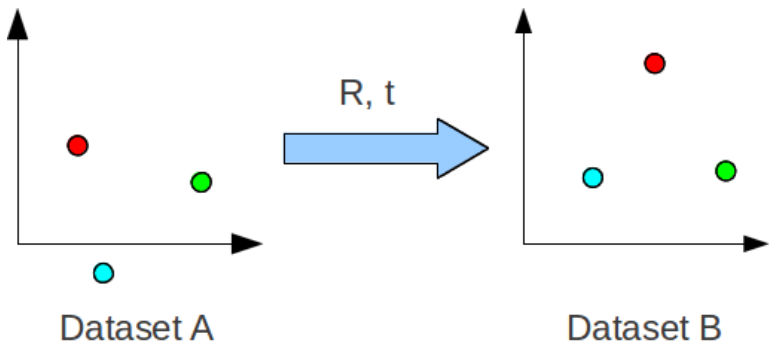

Figure: Rigid transformation.

The MeNet Model for Relative Pose Estimation

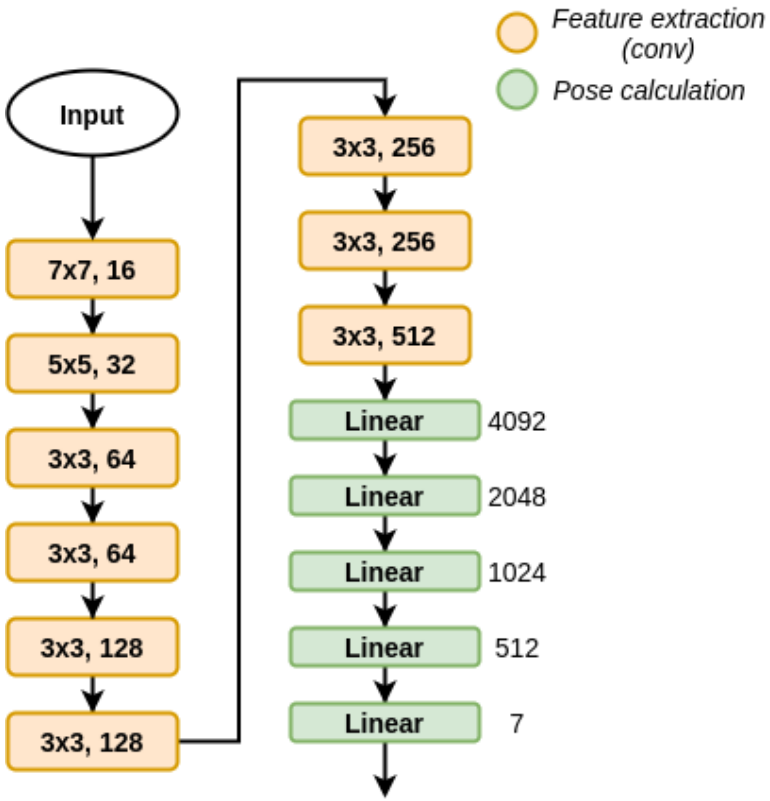

Figure: MeNet model architecture.

The **MeNet** model is targeted for **relative pose estimation**. The input of the network consists in a stack of two images: the goal is to estimate the relative pose of the second image with respect to the first one.

The PoseNet Model for Absolute Pose Estimation

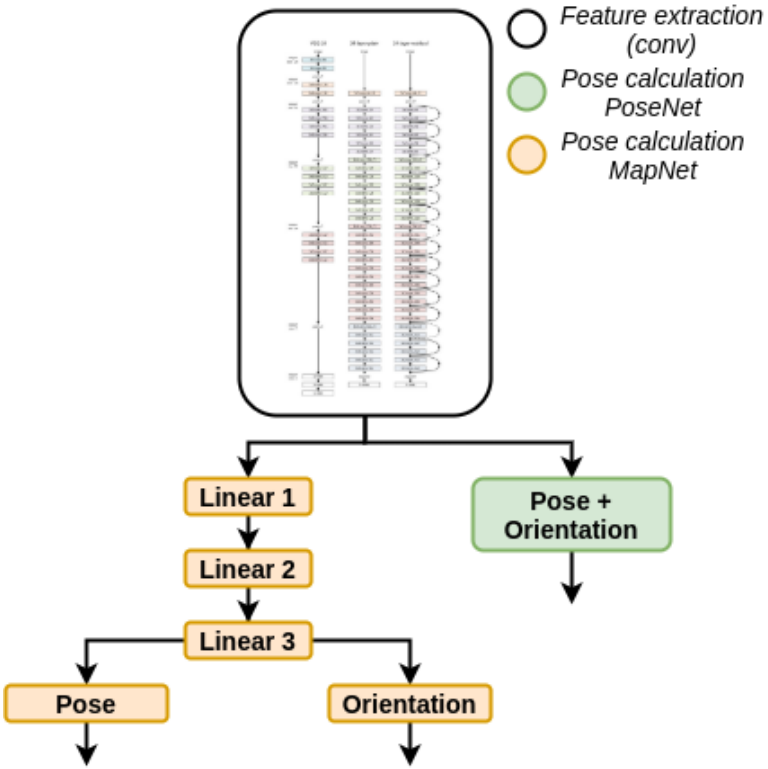

Figure: PoseNet model architecture.

The **PoseNet** model for absolute pose estimation is made up by two components:

- feature extraction through a sequence of convolutional layers (*backend*);
- pose regression on the extracted features using linear layers.

The PoseNet Loss Function

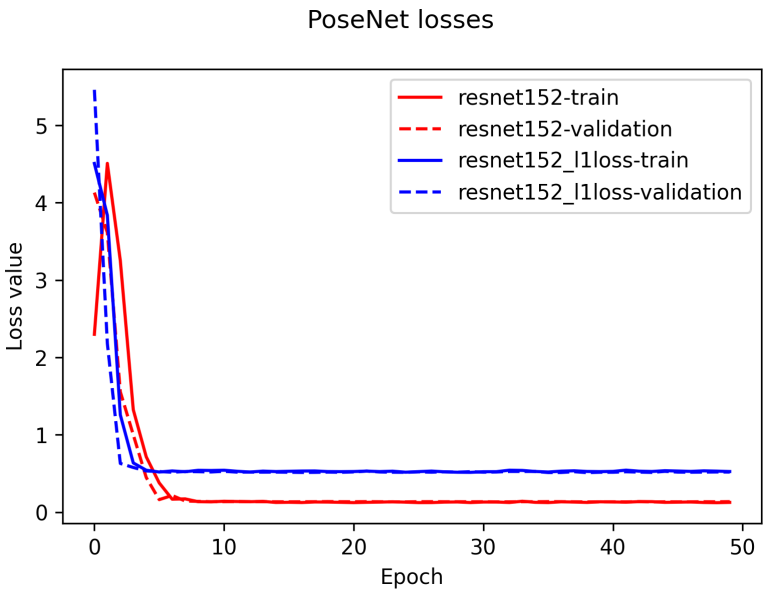

Figure: PoseNet losses trend during training.

Loss	Position Error	Rotation Error
SmoothedL1Loss	**0.594**	**0.139**
L1Loss	0.906	0.226
MSE	NaN	NaN
$\alpha \neq 1$	NaN	NaN

Table: PoseNet losses comparison.

$$Loss(w) = \frac{1}{N}\sum_{i=1}^{N} h(P^i, \hat{P}^i) + \alpha h(Q^i, \hat{Q}^i)$$

PoseNet Results

Figure: PoseNet predictions for our custom dataset. Error $= \sim 2000cm$.

The MapNet Model for Absolute Pose Estimation

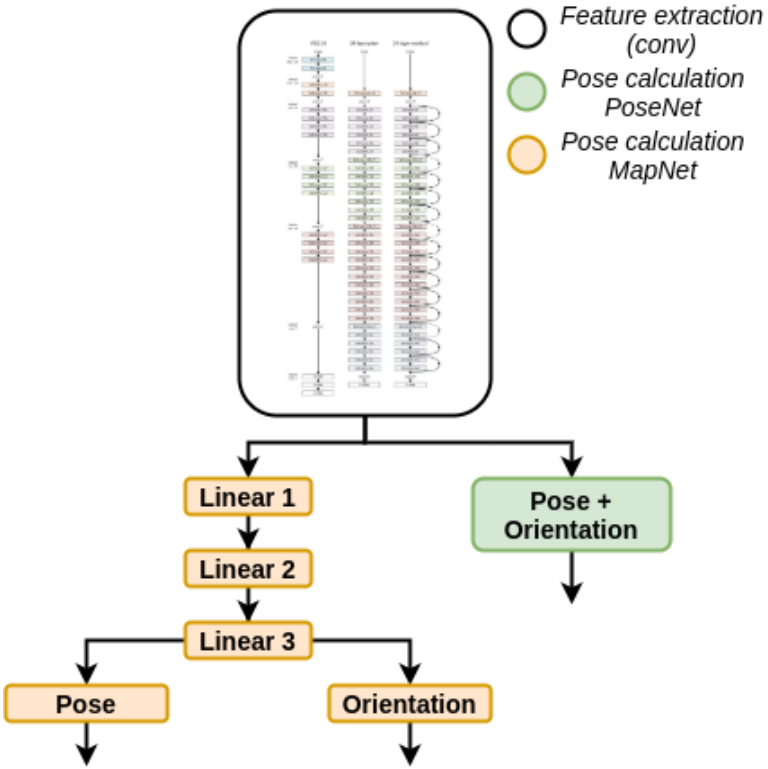

Figure: MapNet model architecture.

The **MapNet** model for absolute pose estimation represents an evolution of the PoseNet model with improvements:

- increase the number of final linear layers;
- penalize both absolute and relative errors in the loss.

The MapNet Loss Function

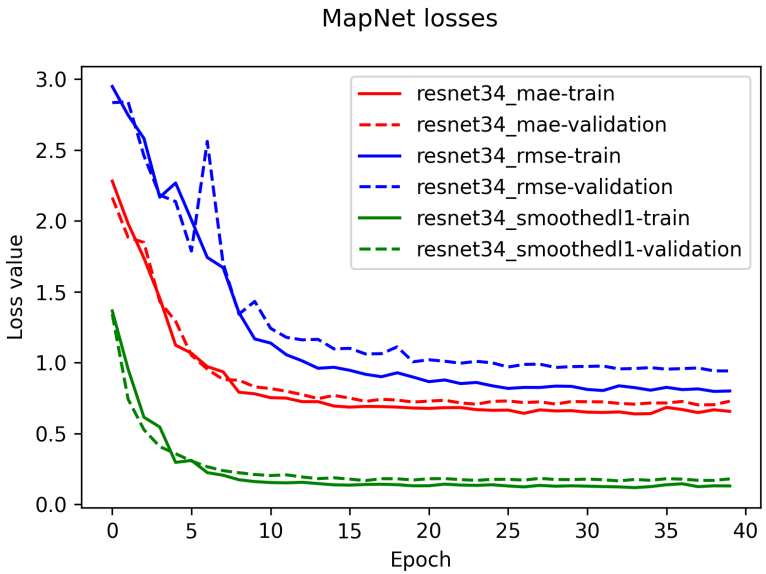

Figure: MapNet losses trend during training.

Loss	Position Error	Rotation Error
L1Loss	0.227	0.042
SmoothedL1Loss	0.187	0.076
RMSE	**0.187**	**0.038**

Table: MapNet losses comparison.

$$L_{\mathcal{D}}(\Theta) = \sum_{i=1}^{|\mathcal{D}|} h([\hat{P}^i \hat{Q}^i], [P^i Q^i])$$

$$+ \alpha \sum_{i,j=1, i \neq j}^{|\mathcal{D}|} h([\hat{P}^{ij} \hat{Q}^{ij}], [P^{ij} Q^{ij}])$$

MapNet Results

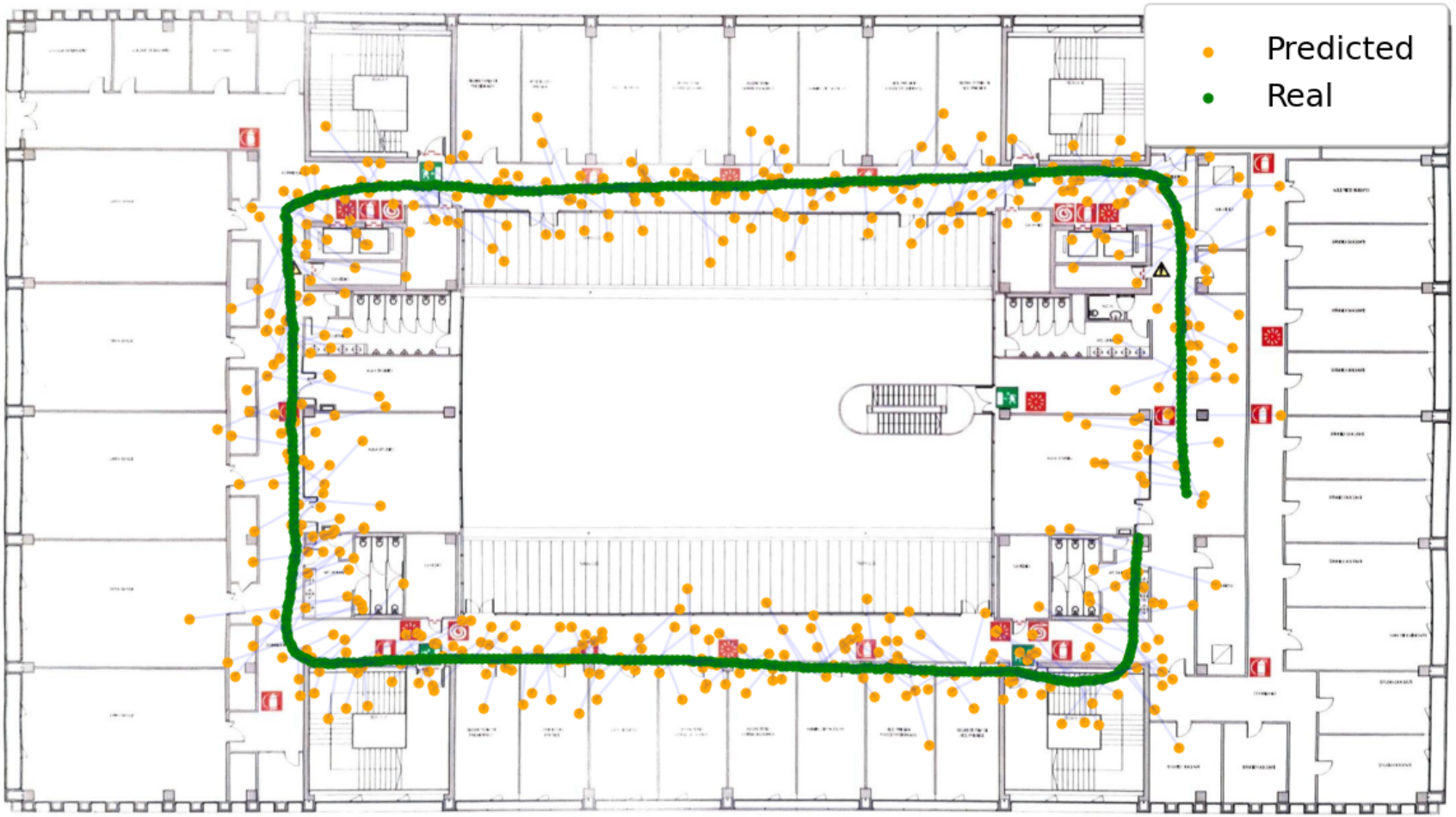

Figure: MapNet predictions for our custom dataset. Error = 153cm.

Detection of Non-Walkable Predictions

We developed post-processing algorithm to improve the model prediction: in case of a non-walkable spot, the output is corrected with the nearest **walkable** point according to the Euclidean distance criterion.

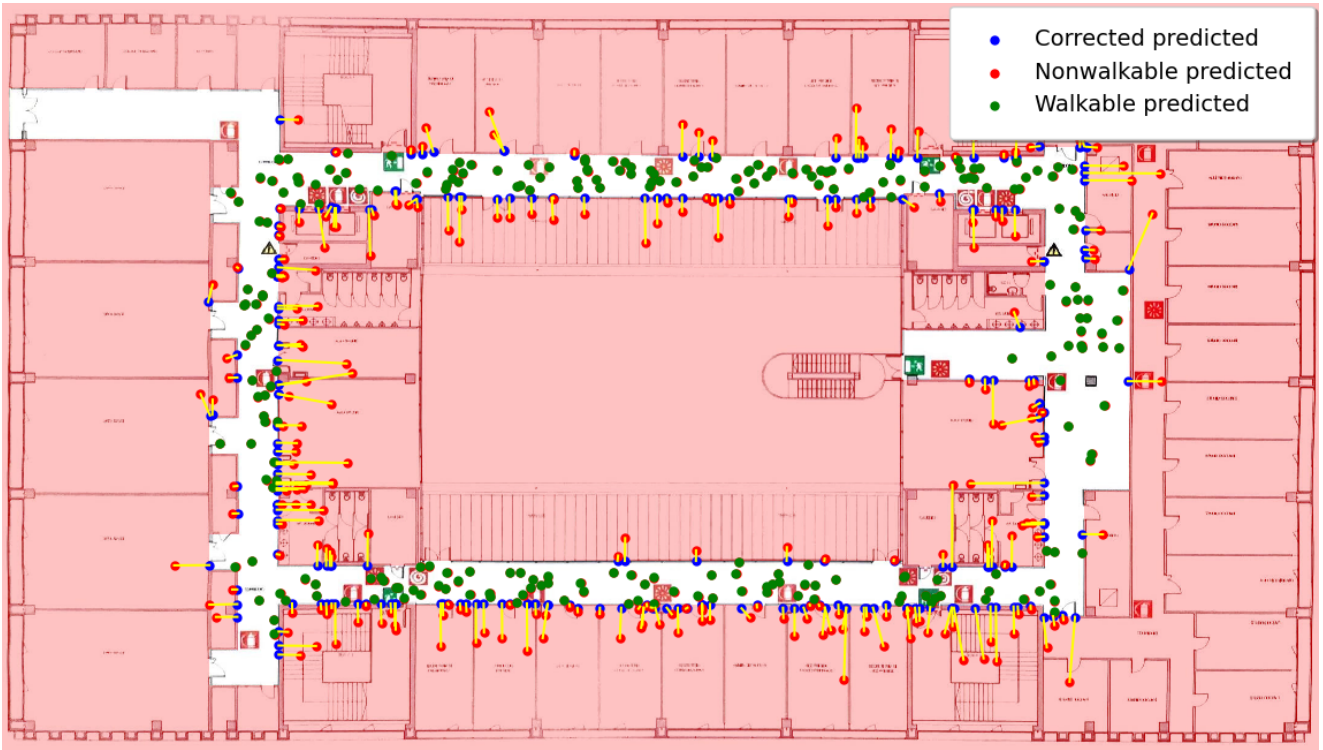

Figure: Prediction map after the post-processing procedure. Error $= 130cm$.

Model Deployment with a Web-Server

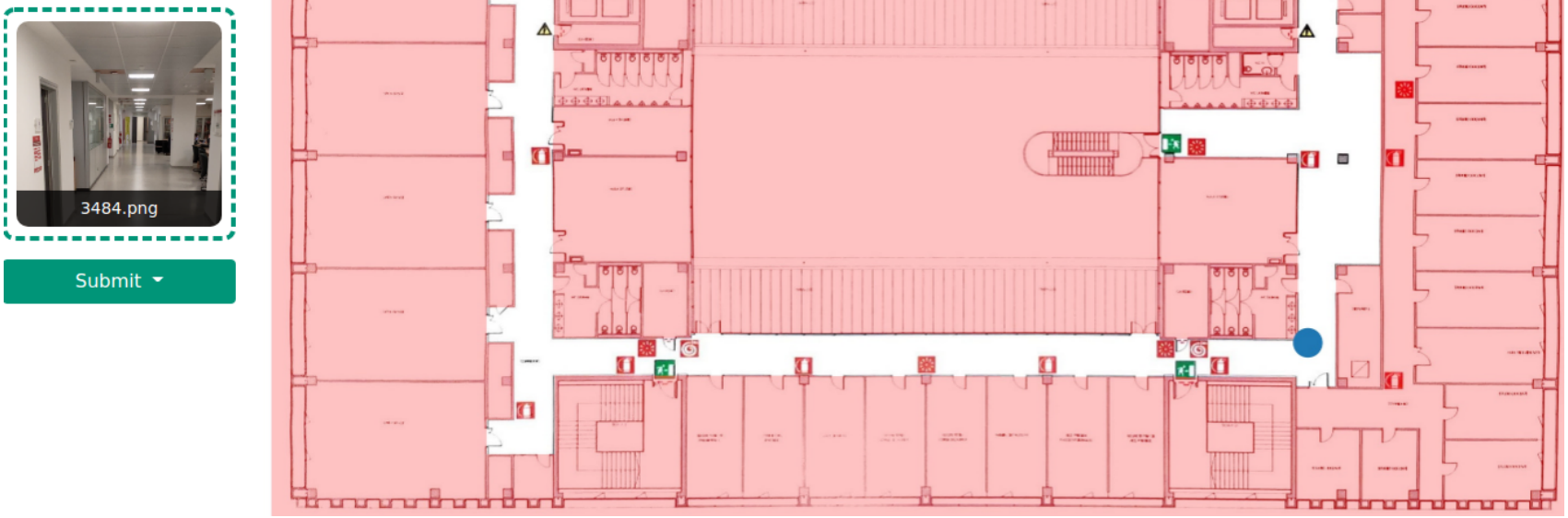

Figure: FastAPI dashboard that serves the model.

Conclusions

To summarize, the final results presented in this work are:

- the exploration of multiple dataset generation techniques;
- the COLMAP reconstruction of Povo 1 second floor;
- the development of relative and absolute pose estimation models;
- the fine-tuning of absolute pose estimation models;
- the post-processing of the model outputs;
- the model deployment using a FastAPI web-server.