Fast vision-based relocalization for MAVs

Quim Sànchez

University of Zurich, Universitat de Girona

June 13, 2014

Table of Contents

Introduction

- New major roles for MAVs
- Operations in GPS denied environments
- Semi-direct Visual Odometry [1]
  - 300 frames per second

Problem

Tracking failure

- Rapid camera motions
- Image occlusion
- Motion blur

A fast and accurate relocalization method is important.

Scenario

- Area recognition. Training stage.
- Tracking is lost. Drone makes a flip.
- Fast 6 DoF relocalization.
  - Provide the pose after relocalization
  - SVO can accept it or not

Implementation

- Independent, extendible, no dependencies from SVO
- Fast, C++
- ROS

Outline

PTAM method

- PTAM is based on keyframes
- The PTAM [2] relocalization method has two main steps
  - **Place Finder**: Given a new frame, it finds the closest keyframe
  - **Relative Pose Finder**: Once the closest frame is found, its transformation to the new frame is calculated

Outline

PTAM Place Finder

The squared difference between *Small Blurry image* is taken as a measure of distance. Cross Correlation distance.

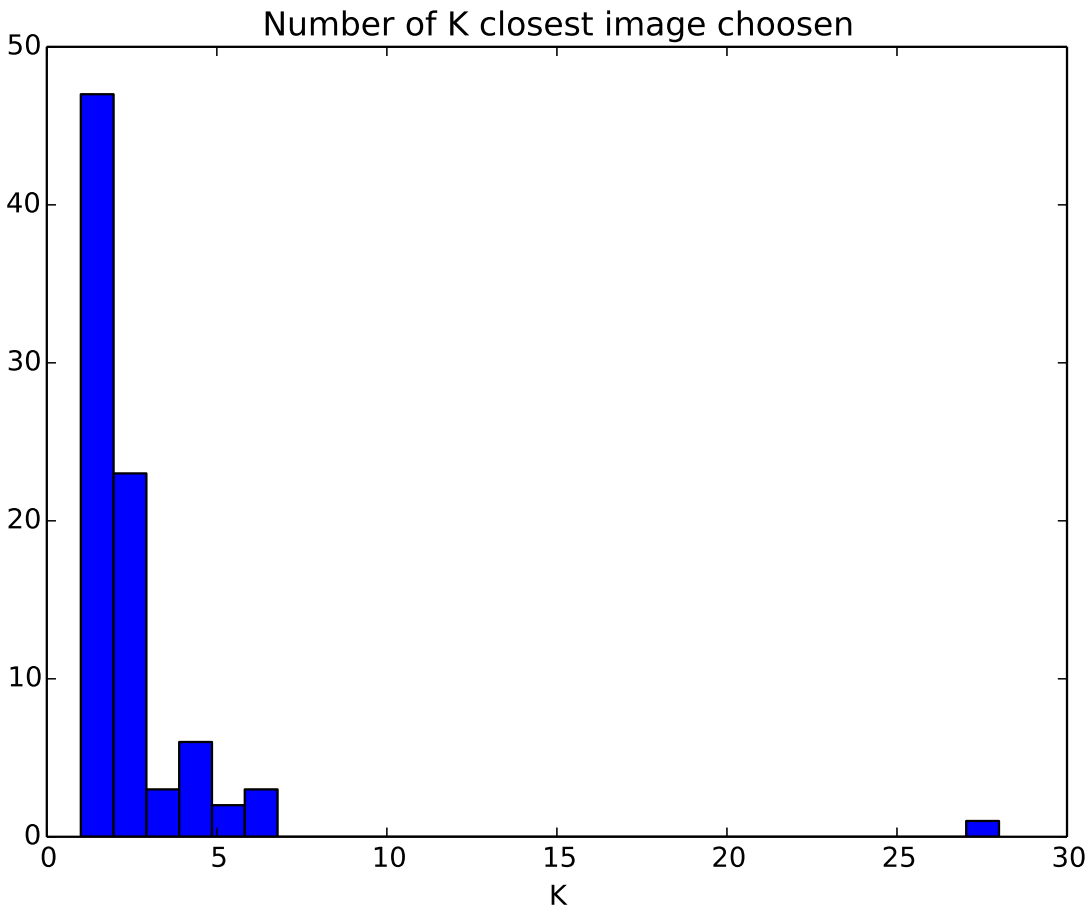
Small Blurry image

- Size 40×30
- Zero mean
- Blurred with Gaussian $\sigma = 2.5$

$$d_{CC} = \sum_{x,y} [(I(x,y) - \bar{I}) - (G(x,y) - \bar{G})]^2 \quad (1)$$

PTAM Place Finder

The squared difference between *Small Blurry image* is taken as a measure of distance. Cross Correlation distance.



Outline

PTAM Rrelative Pose Finder

Image Alignment

The found keyframe and the new image are aligned over translation and rotation, $SE(2)$. Extended Second-Order Minimization (ESM) [3] is used.

PTAM Rrelative Pose Finder

Image Alignment

The found keyframe and the new image are aligned over translation and rotation, $SE(2)$. Extended Second-Order Minimization (ESM) [3] is used.

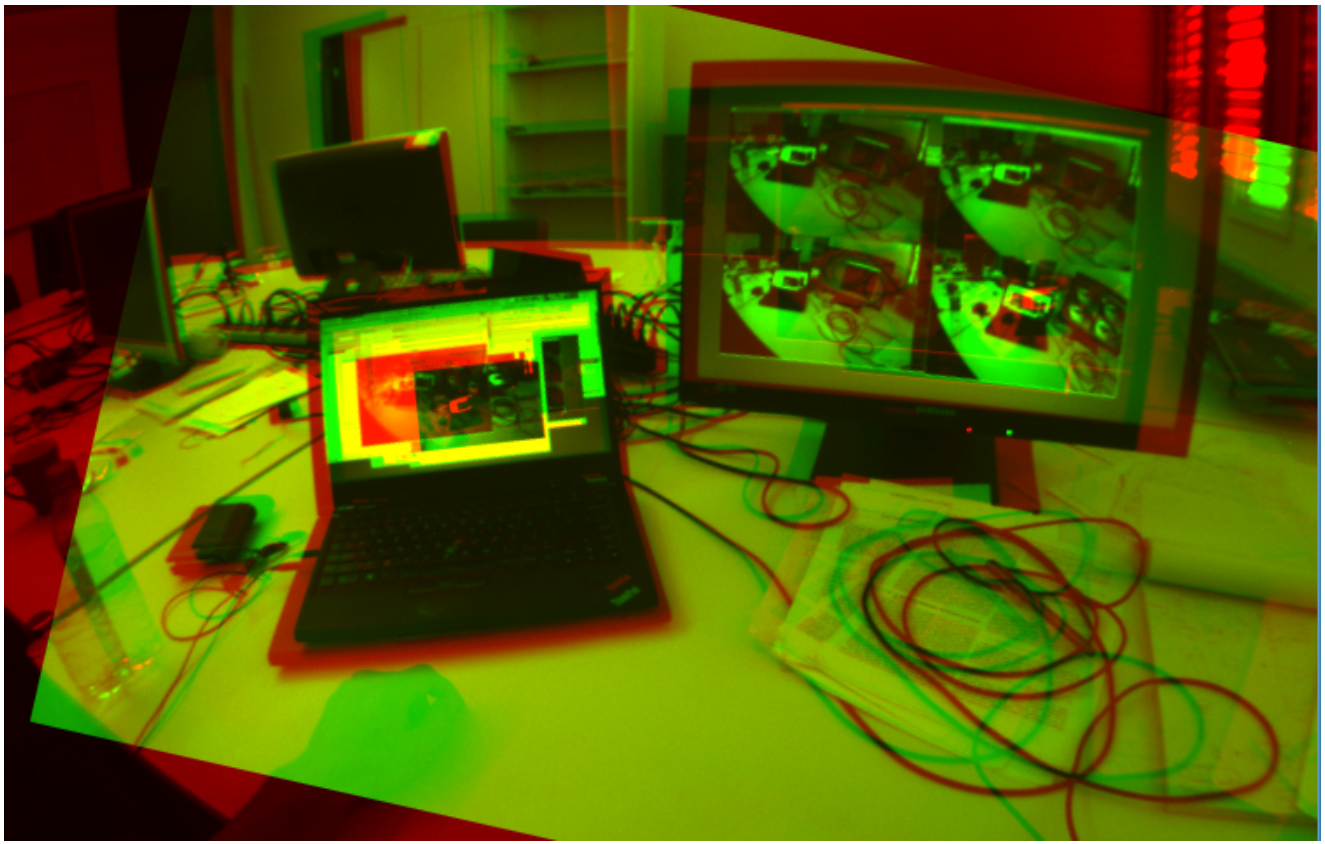
$$W(x;p) = \begin{bmatrix} cos(\alpha) & sin(\alpha) & t_x \\ -sin(\alpha) & cos(\alpha) & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} xcos(\alpha) + ysin(\alpha) + t_x \\ -xsin(\alpha) + ycos(\alpha) + t_y \end{bmatrix} \quad (1)$$

$$\min_p \sum_x [I(W(x;p)) - T(x)]^2 \quad (2)$$

PTAM Rrelative Pose Finder

Image Alignment

The found keyframe and the new image are aligned over translation and rotation, $SE(2)$. Extended Second-Order Minimization (ESM) [3] is used.



3D Interpretations

- The found transformation can be interpreted in different ways from the world frame.
- It is interpreted as world frame rotations, $SO(3)$.
- An optimization framework is used to find the best approximations.

3D Interpretations

- The found transformation can be interpreted in different ways from the world frame.
- It is interpreted as world frame rotations, $SO(3)$.
- An optimization framework is used to find the best approximations.

$$\delta_i(\xi) = T_{SE(2)} u_i - \pi(T_{SO(3)}(\xi) p_i) \qquad \text{where} \quad u_i = \pi(p_i) \qquad (1)$$

3D Interpretations

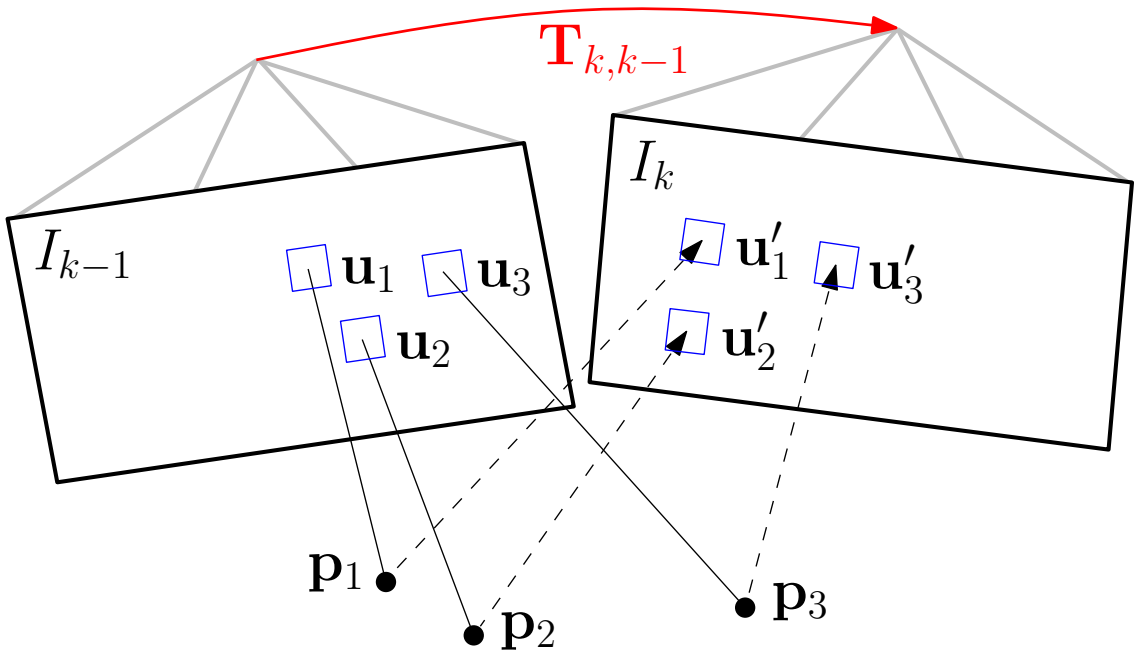
- The found transformation can be interpreted in different ways from the world frame.
- It is interpreted as world frame rotations, $SO(3)$.
- An optimization framework is used to find the best approximations.

Figure: $SE(2)$ transformed image. $SO(3)$ transformed image

Outline

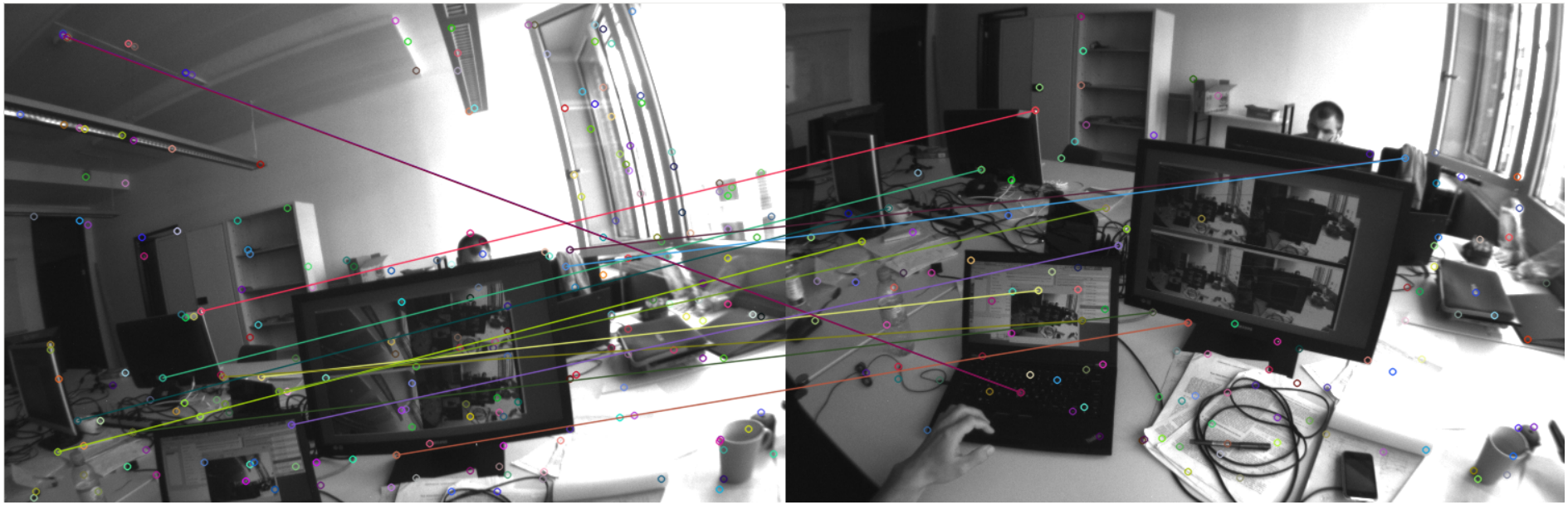
Geometric Relative Pose Finder

- Every keyframe has a world frame point associated to every featured point



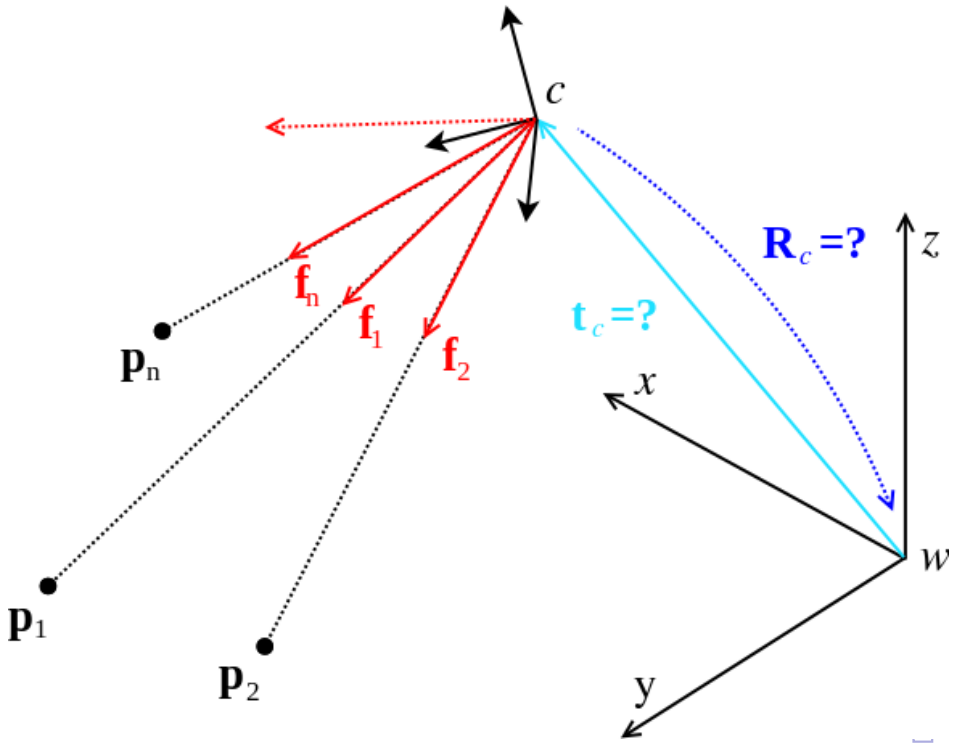
Geometric Relative Pose Finder

- Every keyframe has a world frame point associated to every featured point
- Relate featured points from the new frame to featured points from the keyframe



Geometric Relative Pose Finder

- Every keyframe has a world frame point associated to every featured point
- Relate featured points from the new frame to featured points from the keyframe
- Use P3P to find the camera pose



Outline

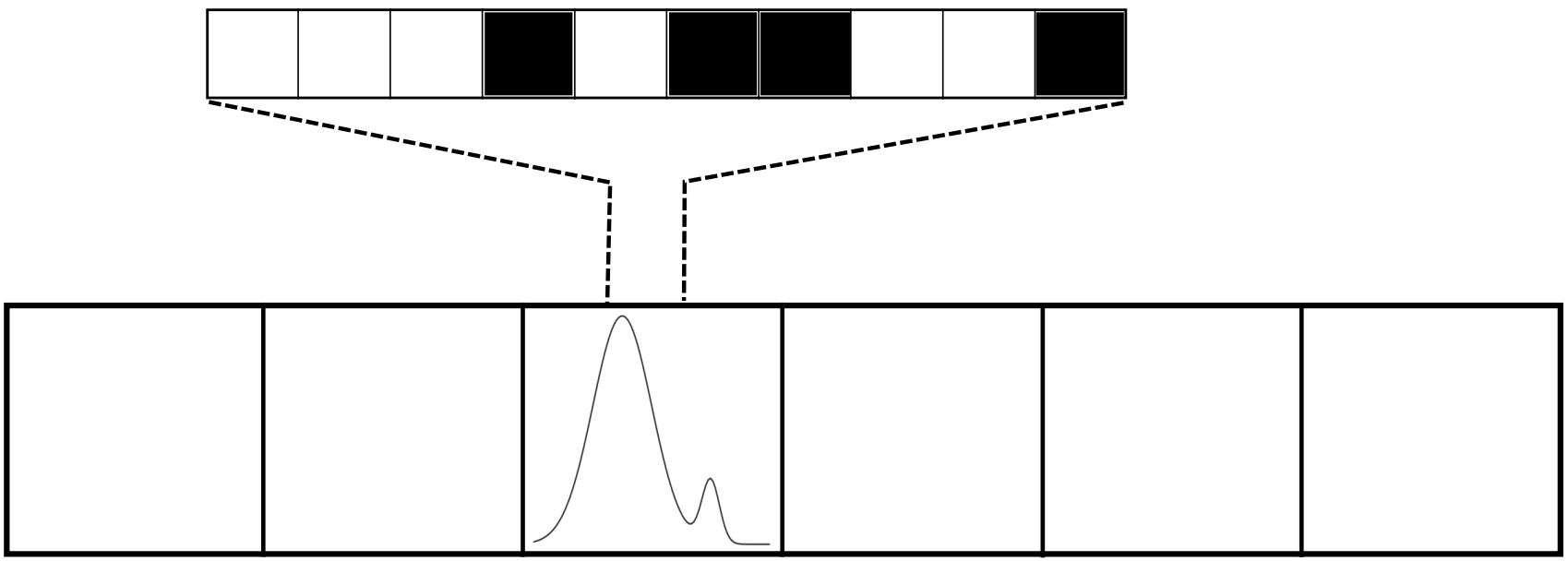
Ferns relocalizer

- Use classifier to model every world frame point with all its views
- Every world frame point is a class
- Multiple views of the point are used to train
  - Artificially generated views are used as well
- Finally, P3P is applied to the found points to find the camera transformation

Ferns relocalizer

- A *fern* [4] is a set of randomly generated binary tests on a patch

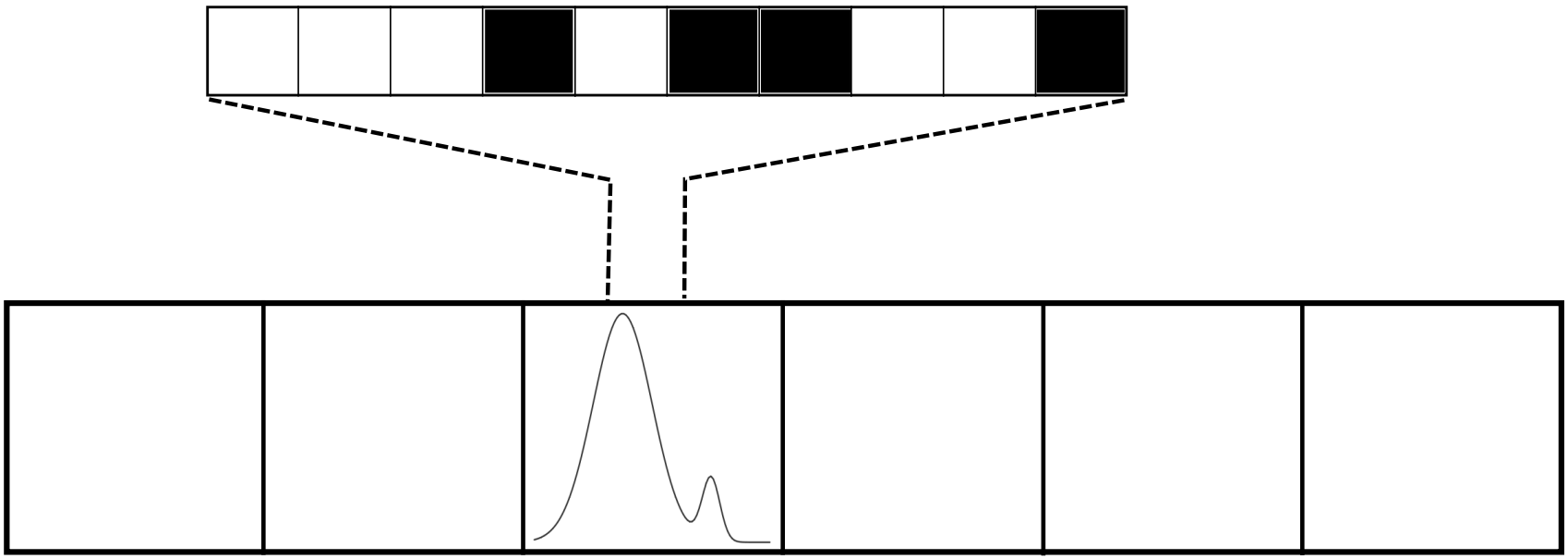
$$f_j = \begin{cases} 1, & \text{if } I(d_j, 1) < I(d_j, 2) \\ 0, & \text{otherwise} \end{cases} \quad (1)$$



Ferns relocalizer

- A *fern* [4] is a set of randomly generated binary tests on a patch
- Every *fern* evaluation refers to a class posterior probability distribution

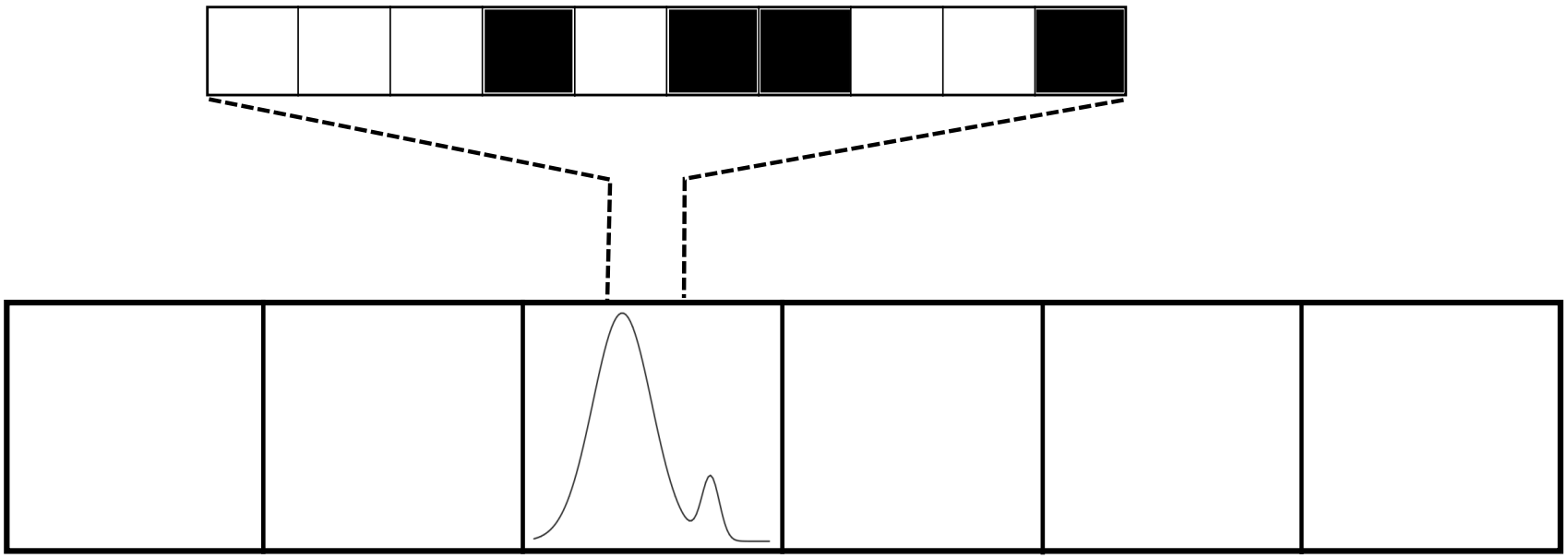
$$f_j = \begin{cases} 1, & \text{if } I(d_j,1) < I(d_j,2) \\ 0, & \text{otherwise} \end{cases} \quad (1)$$



Ferns relocalizer

- A *fern* [4] is a set of randomly generated binary tests on a patch
- Every *fern* evaluation refers to a class posterior probability distribution
- Many *ferns* are used in a classifier

$$f_j = \begin{cases} 1, & \text{if } I(d_j,1) < I(d_j,2) \\ 0, & \text{otherwise} \end{cases} \quad (1)$$



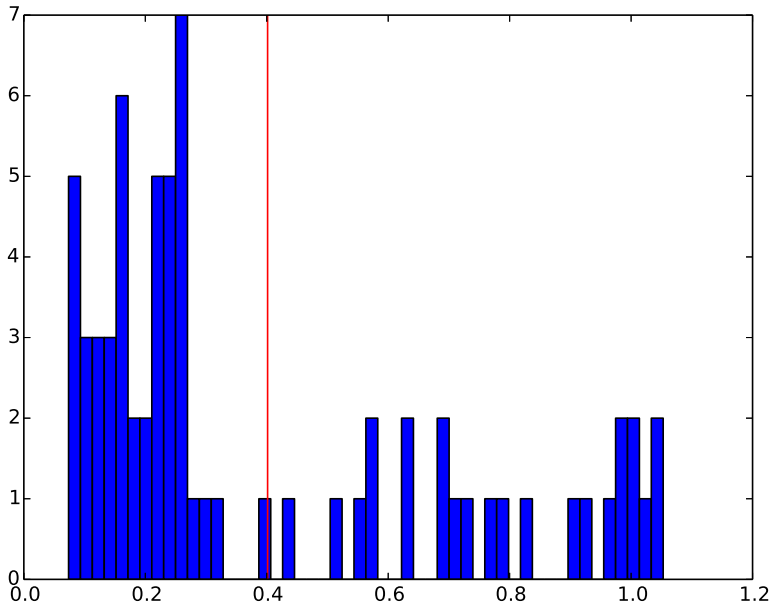
Outline

Test Description

- Training set and testing set
  - Initialize with training set
  - Try to relocalize frames in testing set
- Ground Truth Baseline
  - Use GT data to find optimal Place Finder
- Performance showed histogram of translation error

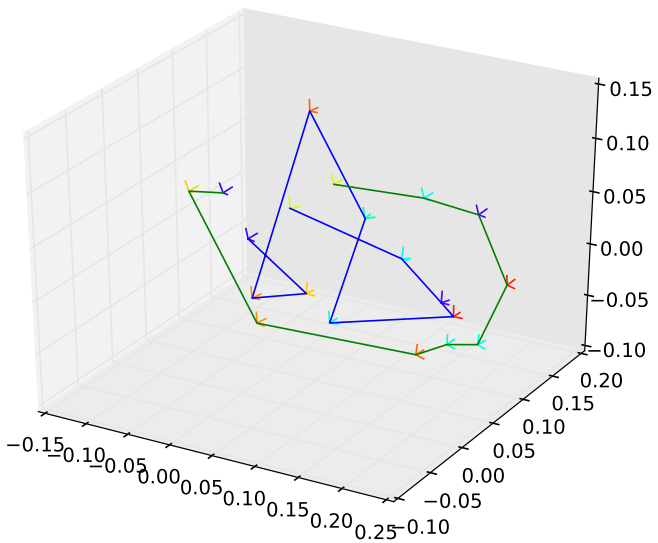
Test Description

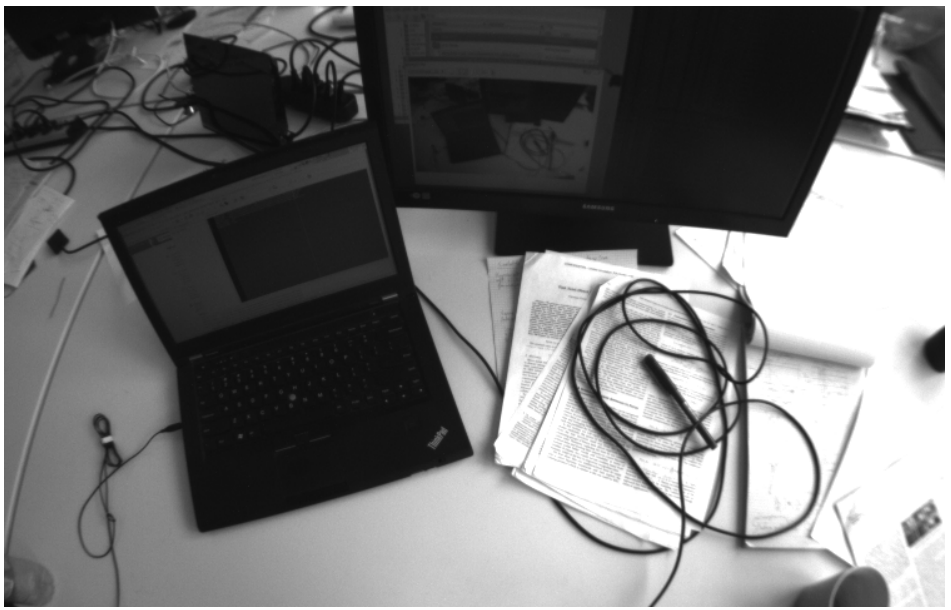
- Training set and testing set
  - Initialize with training set
  - Try to relocalize frames in testing set
- Ground Truth Baseline
  - Use GT data to find optimal Place Finder
- Performance showed histogram of translation error



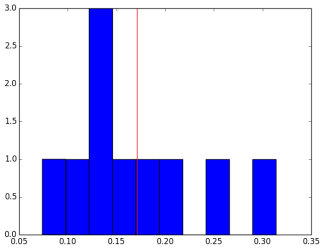
Results small dataset

- 10 images for training and 10 images for testing
- 237 world points
- 2 square meters area

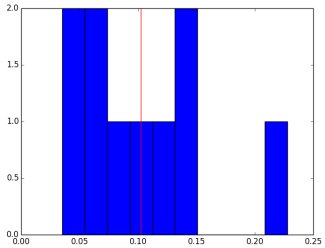




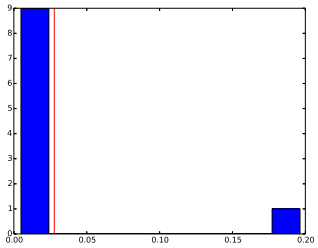
Results small dataset



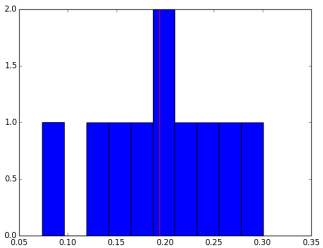
Only GT mean error = 0.17



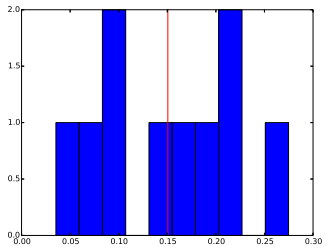
GT and ESM mean error = 0.10



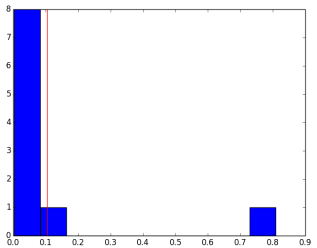
GT and P3P mean error = 0.02



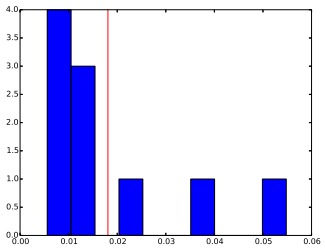
Only CC mean error = 0.19



CC and ESM mean error = 0.15



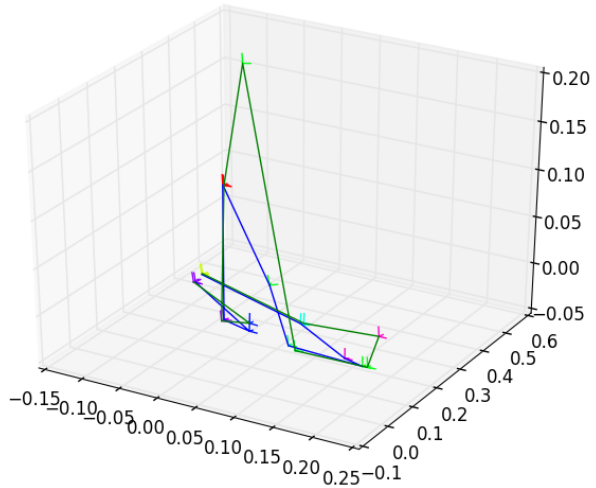
CC and P3P mean error = 0.11



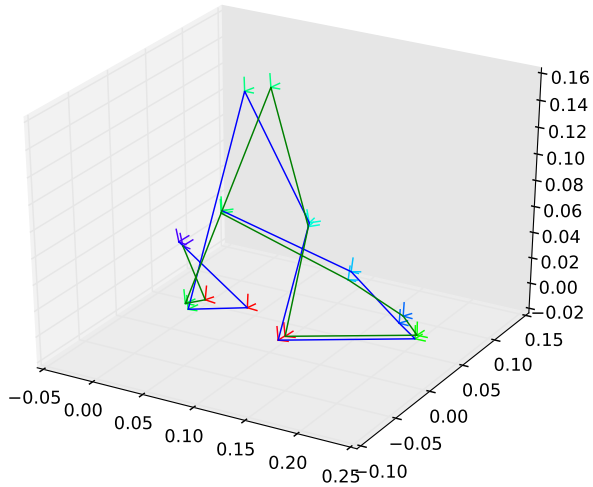
ferns mean error = 0.02

Figure: Translation error histograms

Results small dataset



Path found with CC and P3P

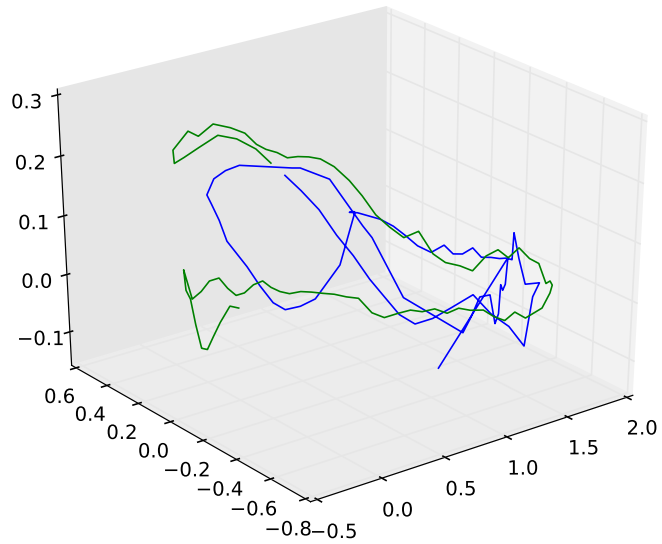


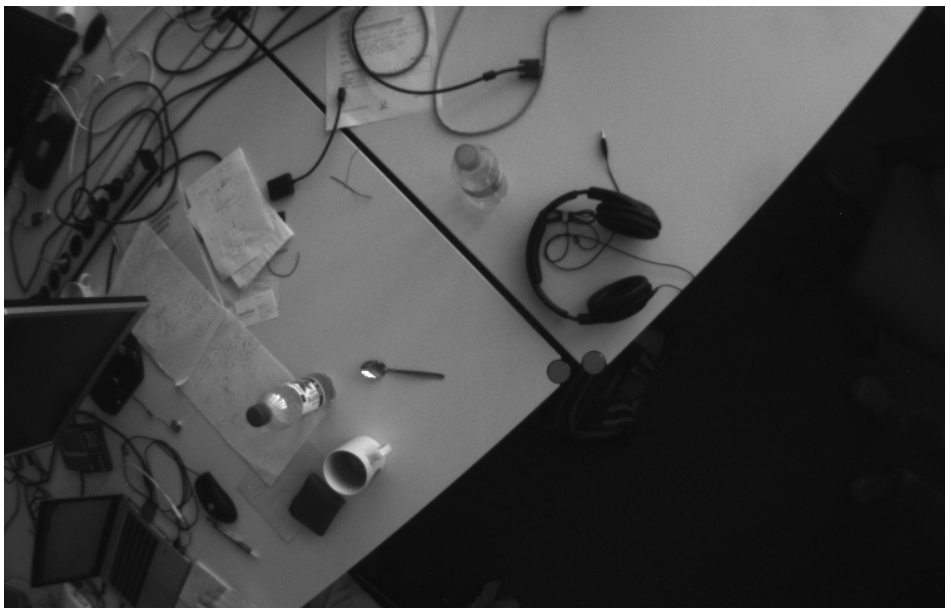
Path found with *ferns*

Outline

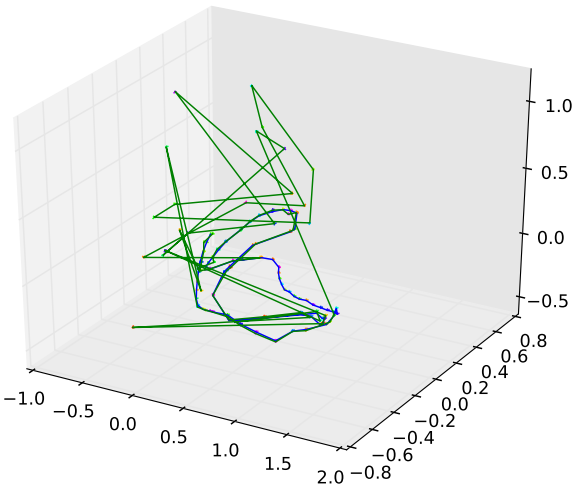
Results large dataset

- 84 images for training and 69 images for testing
- 1730 world points
- 7x3 meters area

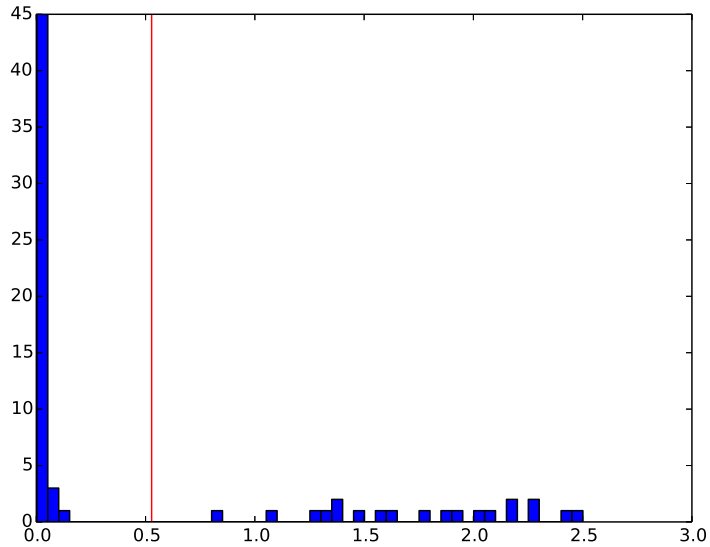




Results large dataset with *ferns*



Path found with *ferns*



ferns error histogram. 45/69

Timing

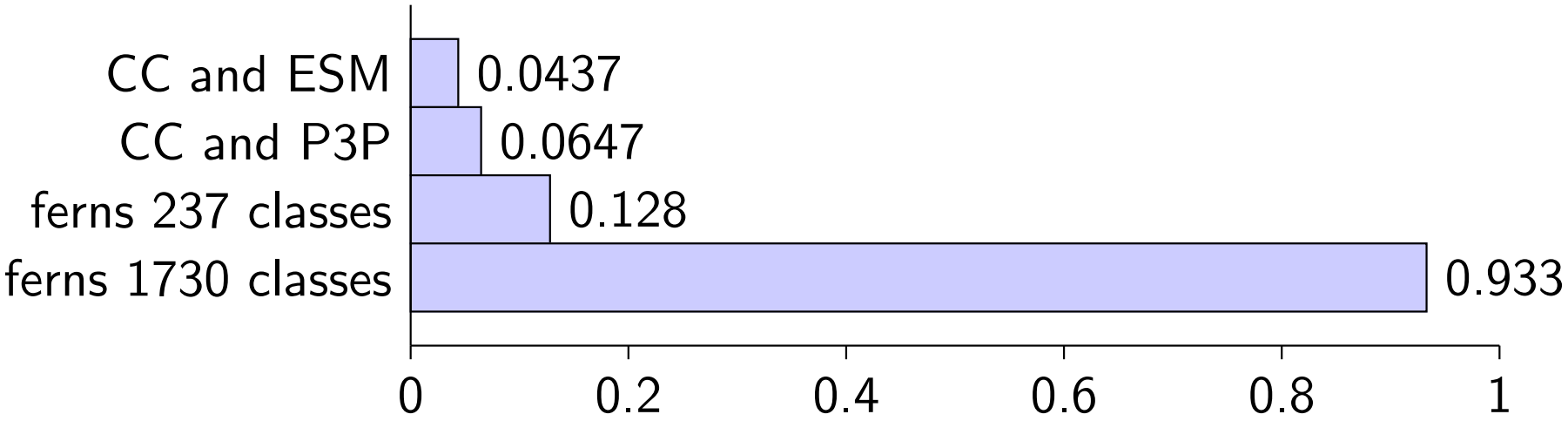


Figure: Single relocalization execution time in seconds

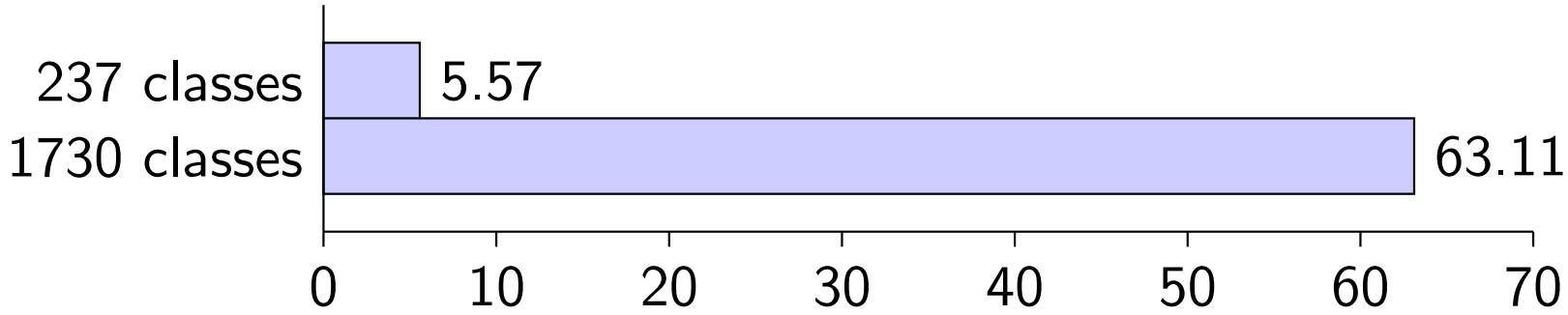


Figure: *ferns* classifier training time in seconds

Outline

Conclusions

Work done

- We have addressed one of the missing parts of SVO
- Different methods have been studied an evaluated
- An independent framework and library have been implemented

Future work

- Further tests on board
- Use image retrieval techniques like bag of word as Place Finder
- Test performance on rectified images

Thank you!

Questions?

Bibliography


C. Forster, M. Pizzoli, and D. Scaramuzza.
SVO: Fast Semi-Direct Monocular Visual Odometry.
Proc. IEEE Intl. Conf. on Robotics, 2014.


G. Klein and D. Murray.
Parallel Tracking and Mapping for Small AR Workspaces.
IEEE and ACM International Symposium on Mixed and Augmented Reality, Nov. 2007.


S. Lovegrove.
Parametric dense visual SLAM.
PhD thesis, Imperial College London, 2012.


M. Ozuysal and M. Calonder.
Fast keypoint recognition using random ferns.
Pattern Analysis, 2010.