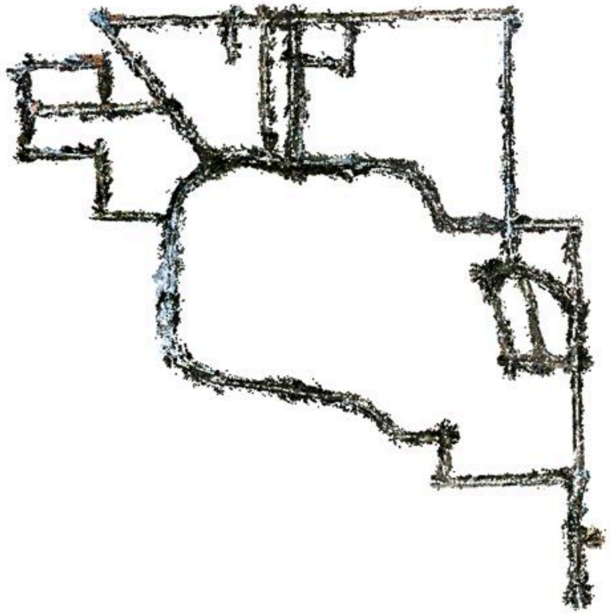
Stereo Visual Odometry

Chris Beall

CVPR 2014 Visual SLAM Tutorial

OUTLINE

- Motivation
- Algorithm Overview
- Feature Extraction and matching
- Incremental Pose Recovery
- Practical/robustness considerations
- What else do you get?
- Results

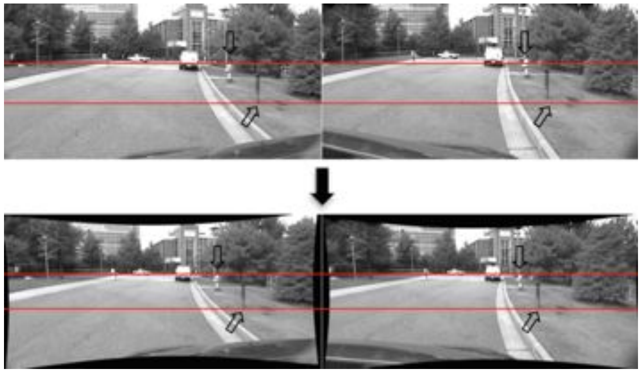


Motivation

- Why stereo Visual Odometry?
  - Stereo avoids scale ambiguity inherent in monocular VO
  - No need for tricky initialization procedure of landmark depth

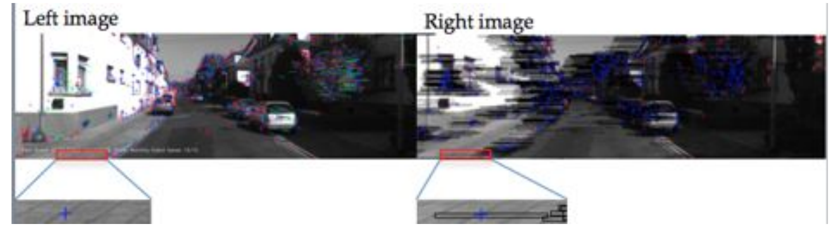
Algorithm Overview

1. Rectification



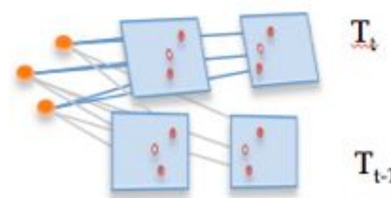
2. Feature Extraction

3. Stereo Feature Matching

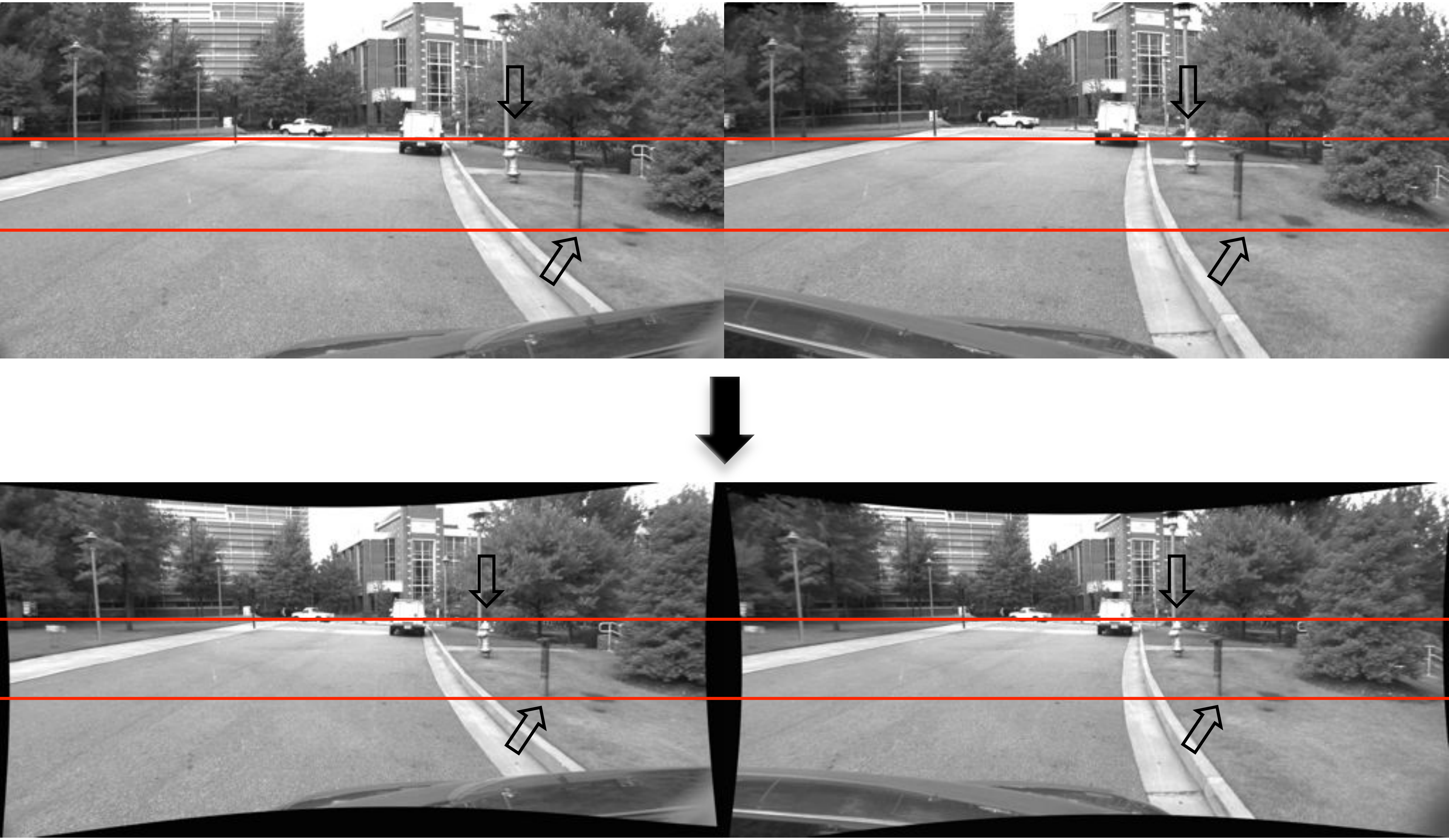


4. Temporal Feature Matching

5. Incremental Pose Recovery/RANSAC

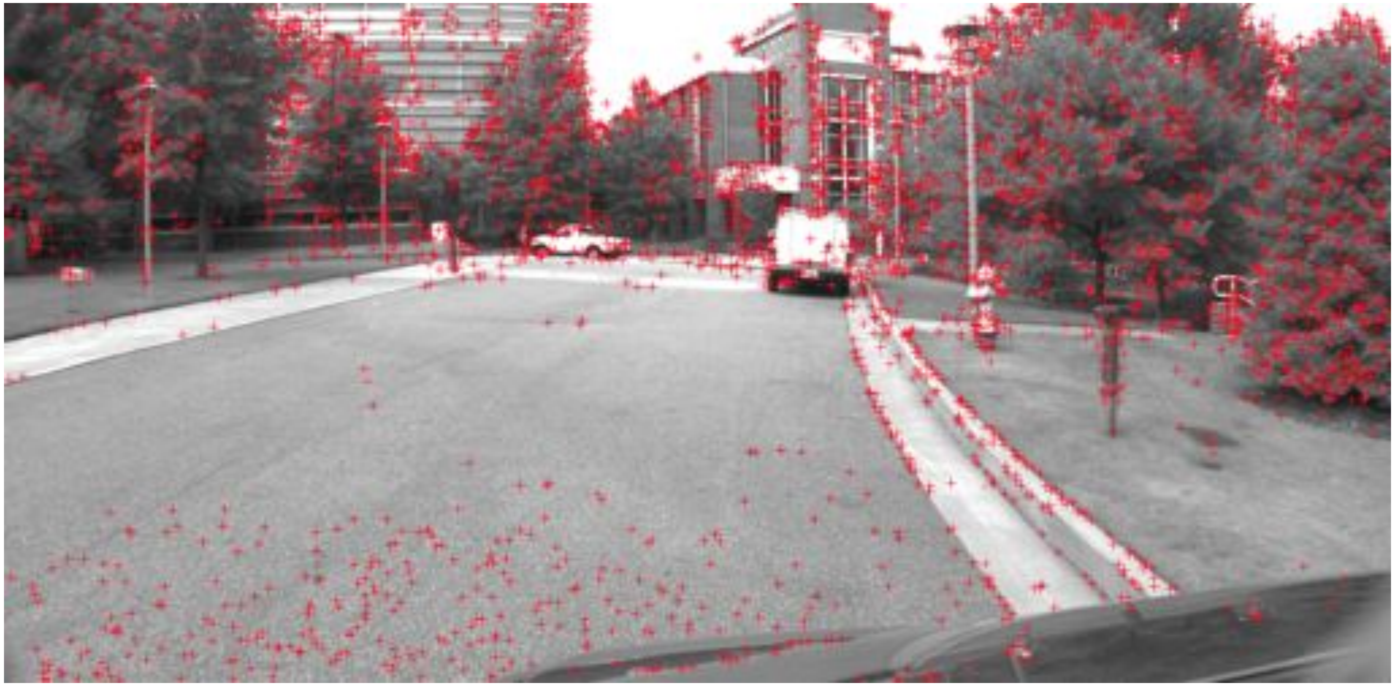


Undistortion and Rectification



Feature Extraction

- Detect local features in each image
  - SIFT gives good results (can also use SURF, Harris, etc.)



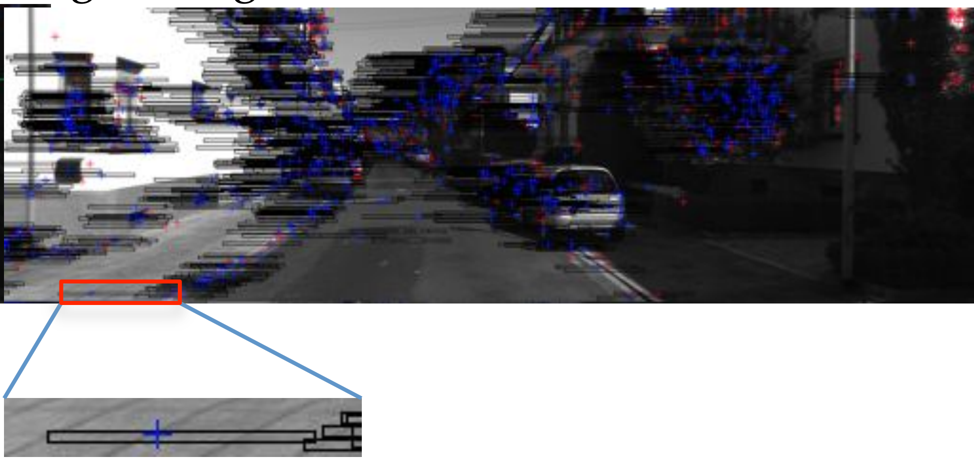
Lowe ICCV 1999

Stereo Matching

- Match features between left/right images
- Since the images are rectified, we can restrict the search to a bounding box on the same scan-line

Left image

Right image



Temporal Matching

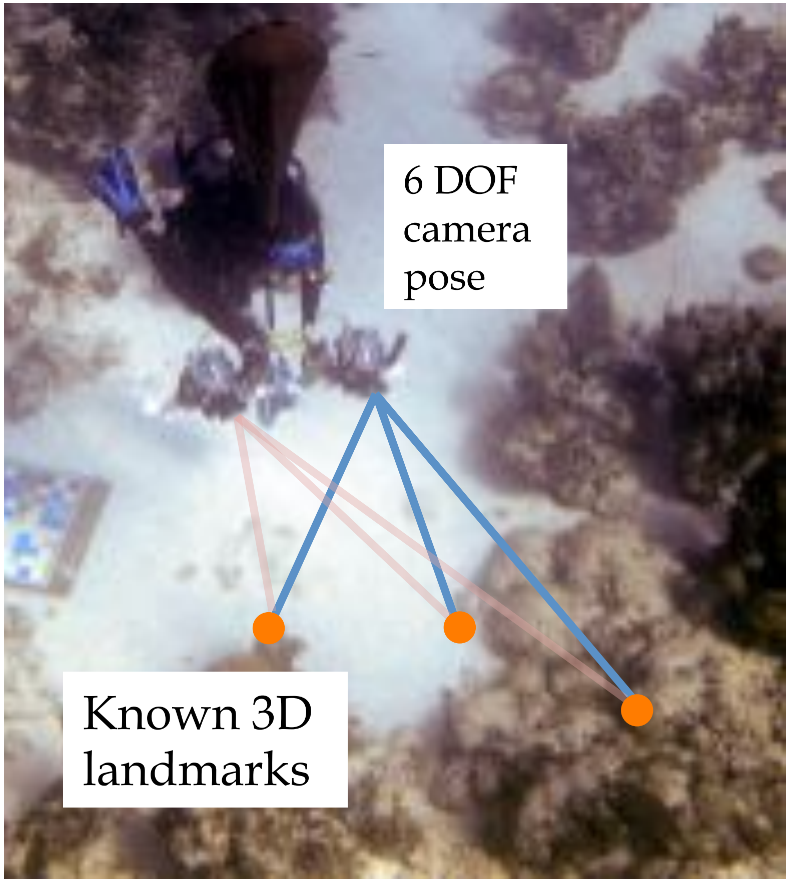
- Temporally match features between frame t and t-1

Relative Pose Estimation/RANSAC

- Want to recover the incremental camera pose using the tracked features and triangulated landmarks
- There will be some erroneous stereo and temporal feature associations → Use RANSAC
  - Select N out of M data items at random (the minimal set here is 3)
  - Estimate parameter (incremental pose from t-1 to t)
  - Find the number K of data items that fit the model (called inliers) within a given tolerance
  - Repeat S times
  - Compute refined model using full inlier set

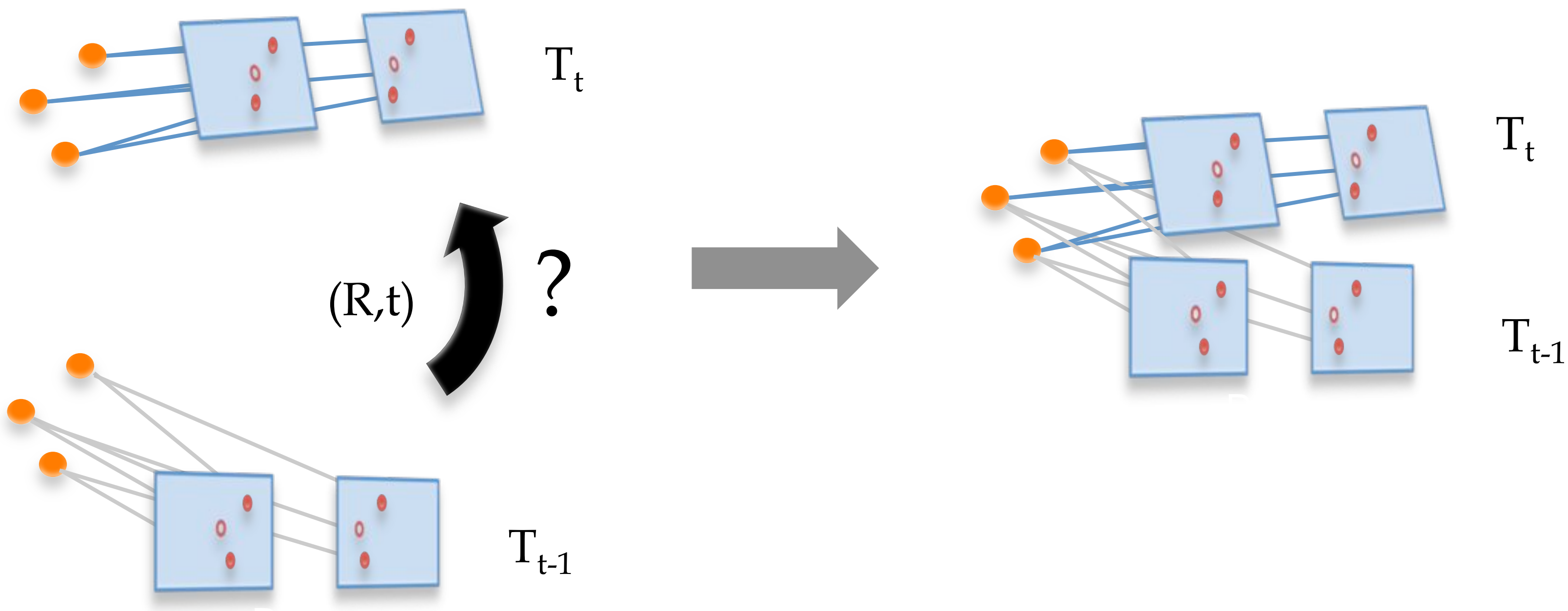
Relative Pose Estimation

- Camera pose can be recovered given at least three known landmarks in a non-degenerate configuration
- In the case of stereo VO, landmarks can simply be triangulated
- Two ways to recover pose:
  - Absolute orientation
  - Reprojection error minimization



Absolute Orientation

- Estimate relative camera motion by computing relative transformation between 3D landmarks which were triangulated from stereo-matched features

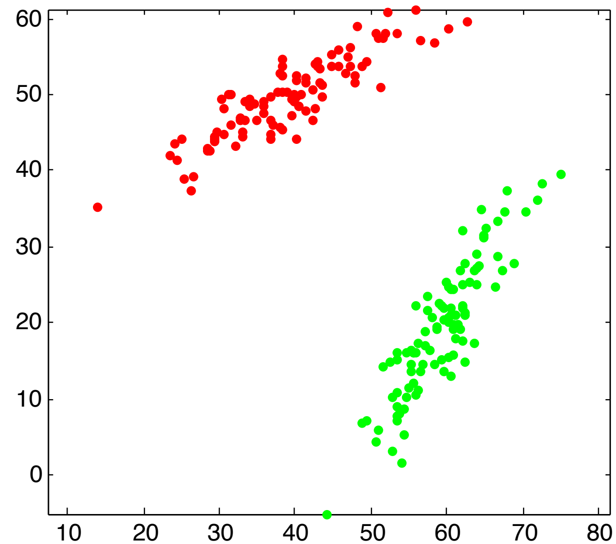


Absolute Orientation

- First, in 2D:
  - Given two sets of corresponding points $\{p_i^l\}$ and $\{p_i^r\}$ related by a rigid 2D transformation $T_r^l = (R_r^l, t_r^l)$:

$$p^l = R_r^l p^r + t_r^l$$

  - First recover rotation (2 points), then translation (1 point)



B. K. P Horn JOSA 1987

Absolute Orientation

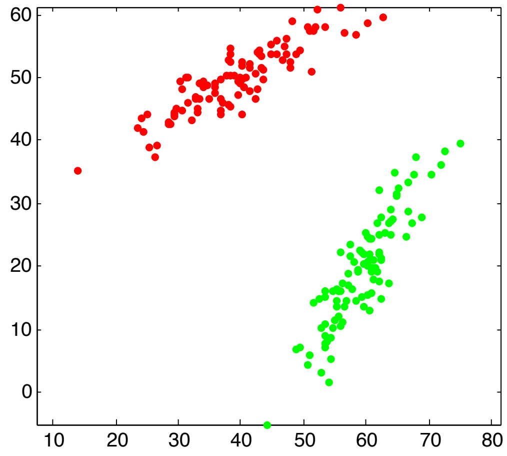
- Rotation:
  - Given two point correspondences (p_1^l, p_1^r) and (p_2^l, p_2^r) , the vector v between the two points will be rotated by the desired angle θ
  - Specfically, the vectors are related by

$$v^l = (p_2^l - p_1^l) = R_r^l \, (p_2^r - p_1^r) = R_r^l v^r$$

  - Finally, recover the angle

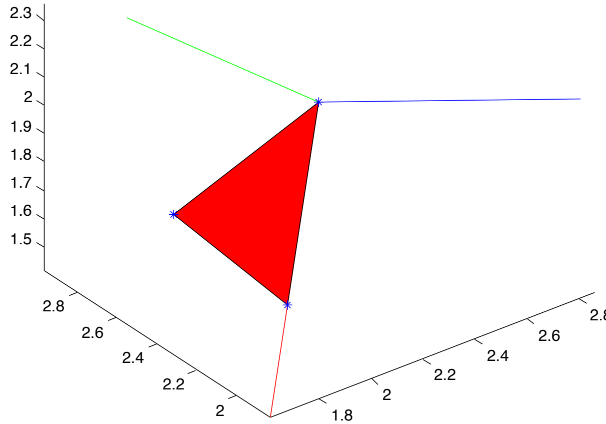
$$\theta = \arccos \frac{v^l \cdot v^r}{\|v^l\| \, \|v^r\|}$$

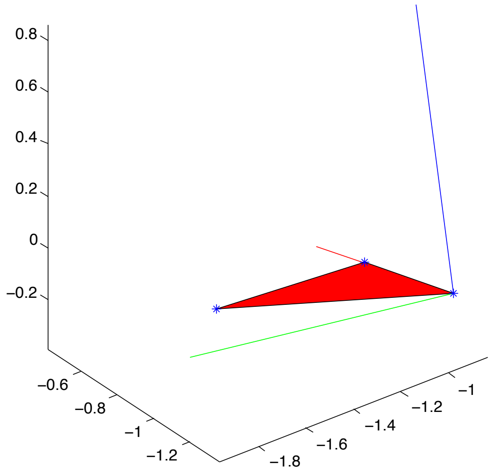
and translation $t_r^l = p^l - R_r^l p^r$



Absolute Orientation

- Now in 3D:
  - Create coordinate frames from three corresponding points
    - Take x-axis by connecting p_1^l to p_2^l : $\hat{x}_l \propto (p_2^l - p_1^l)$
    - Construct y-axis in the plane formed by three points, perpendicular to x-axis: $\hat{y}_l \propto (p_3^l - p_1^l) - \left[(p_3^l - p_1^l) \cdot \hat{x}_l\right] \hat{x}_l$
    - Complete frame with z-axis: $\hat{z}_l \propto (\hat{x}_l \times \hat{y}_l)$





Absolute Orientation

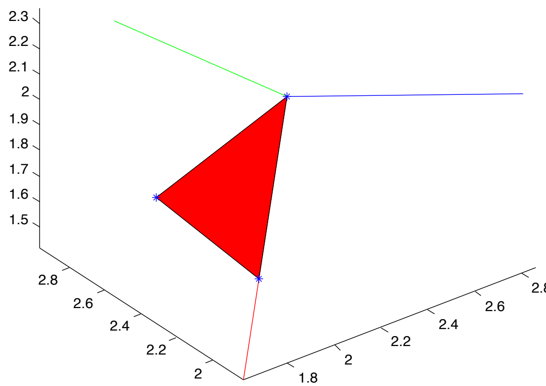
- Normalize the axes and we have the rotation of the frame with respect to the global reference frame,

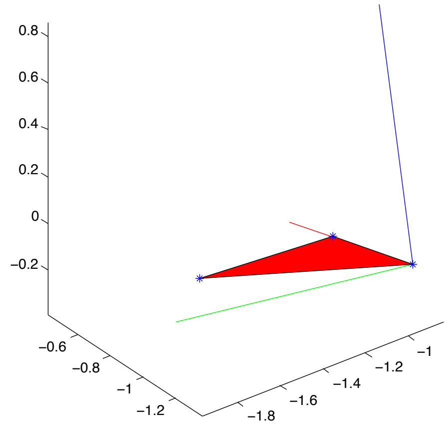
$$R_l^g = \begin{bmatrix} \hat{x}_l & \hat{y}_l & \hat{z}_l \end{bmatrix}$$

- Repeat for the right frame, and obtain the relative rotation,

$$R_r^l = R_g^l R_r^g = \left(R_l^g\right)^T R_r^g$$

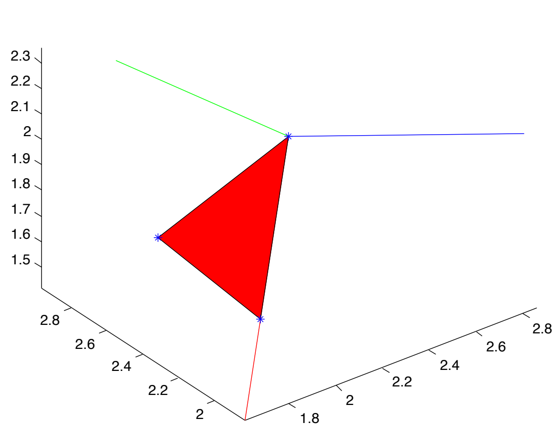
- As in the 2D case, the translation can then be recovered using a single point

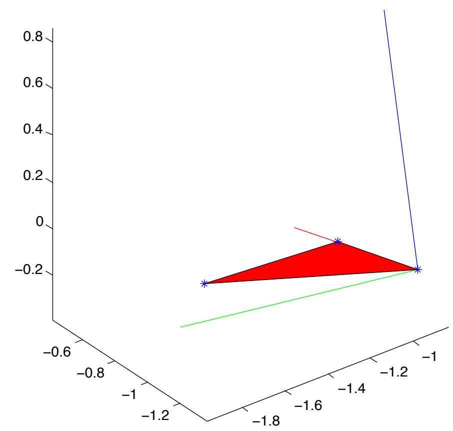




Absolute Orientation

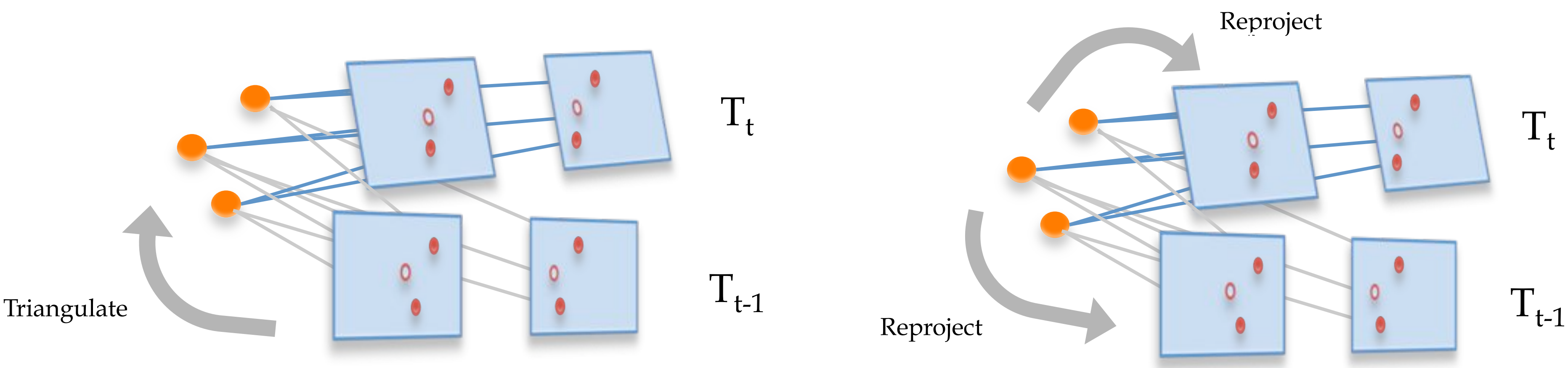
- The Absolute Orientation approach assumes a relatively noiseless case, and does not work well otherwise
- No simple way to average out noisy points by considering more data
  - Use SVD-based method instead
  - Use different approach based on projective geometry





Reprojection Error Minimization

- A better approach is to estimate relative pose by minimizing reprojection error of 3D landmarks into images at time t and t-1



Practical/Robustness Considerations

- The presented algorithm works very well in feature-rich, static environments, but…
- A few tricks for better results in challenging conditions:
  - Feature binning to cope with bias due to uneven feature distributions
  - Keyframing to cope with dynamic scenes, as well as reducing drift of a stationary camera
- Real-time performance

Feature Binning

- Incremental pose estimation yields poor results when features are concentrated in one area of the image
- Solution: Draw a grid and keep k strongest features in each cell

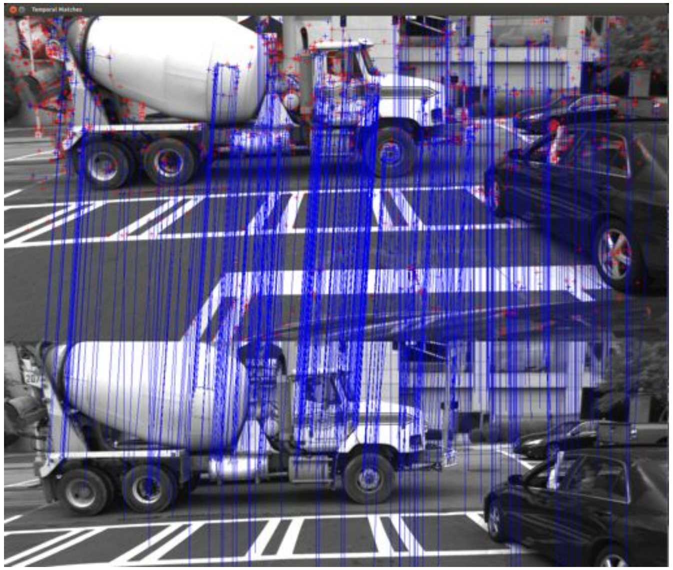
Challenges: Dynamic Scenes

- Dynamic, crowded scenes present a real challenge
- Cannot depend on RANSAC to always recover the correct inlier set
- Example 1: Large van "steals" inlier set in passing

Inliers Outliers

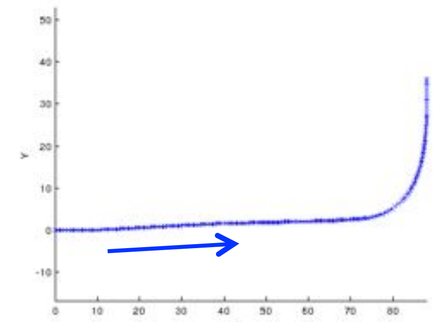
Challenges: Dynamic Scenes

- Example 2: Cross-traffic while waiting to turn left at light



Only accept incremental pose if:
- translation > 0.5m
- Dominant direction is forward

Incorrect sideways motion

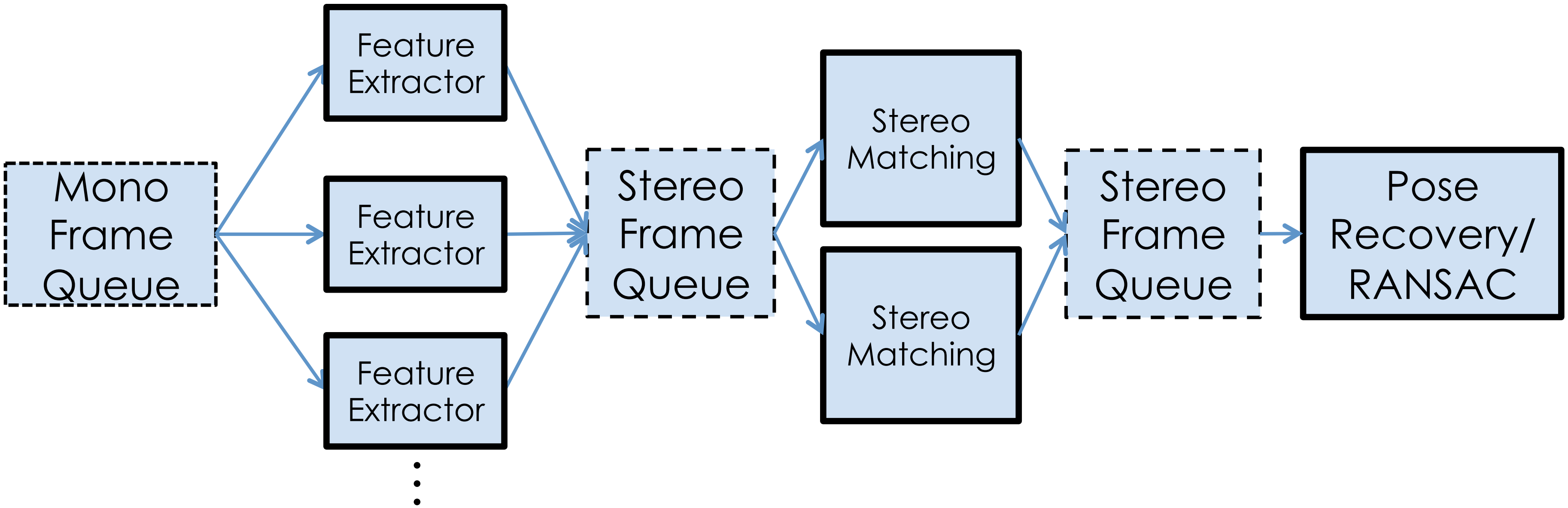


Without keyframing

With keyframing

Real-time performance

- Parallelization of feature extraction and stereo matching steps allows real-time performance even in CPU-only implementation

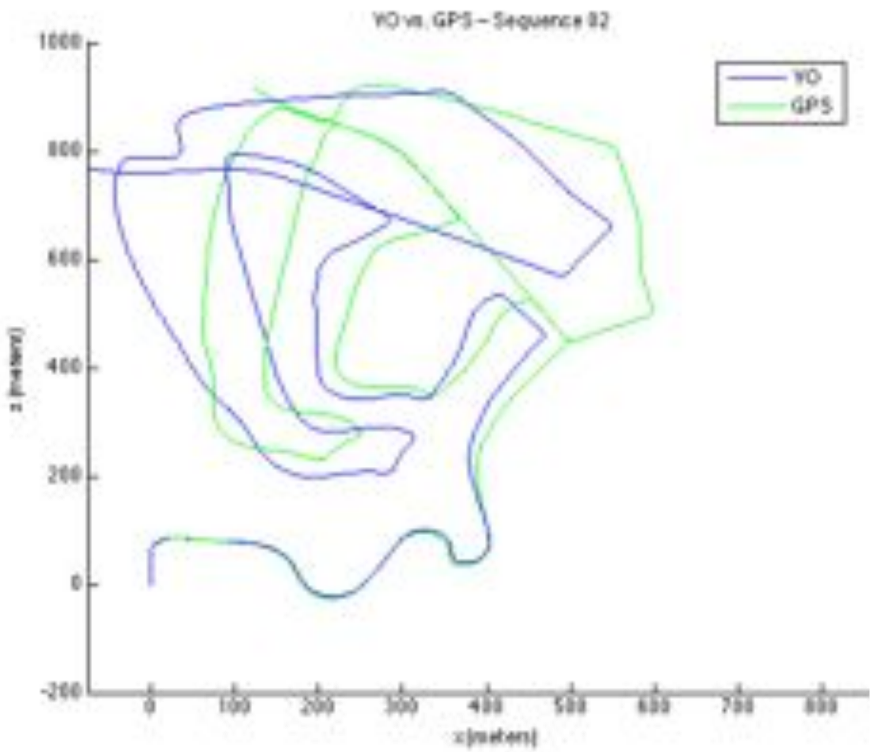


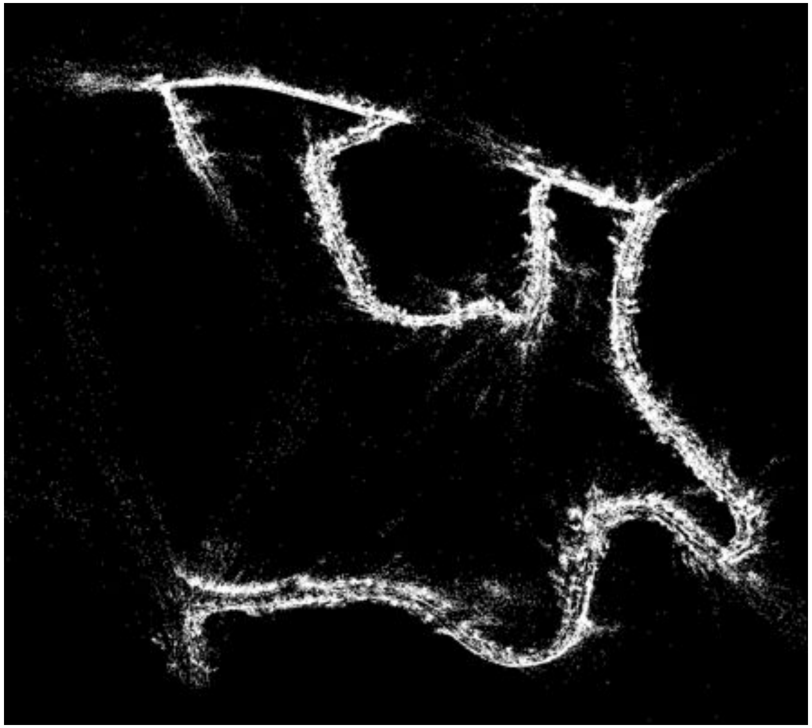
What else do you get?

- Stereo Visual Odometry yields more than just a camera trajectory!
- Tracked landmarks form a sparse 3D point cloud of the environment
  - Can be used as the basis for localization

What else do you get?

- 3D Point cloud on KITTI Benchmark, Sequence 2

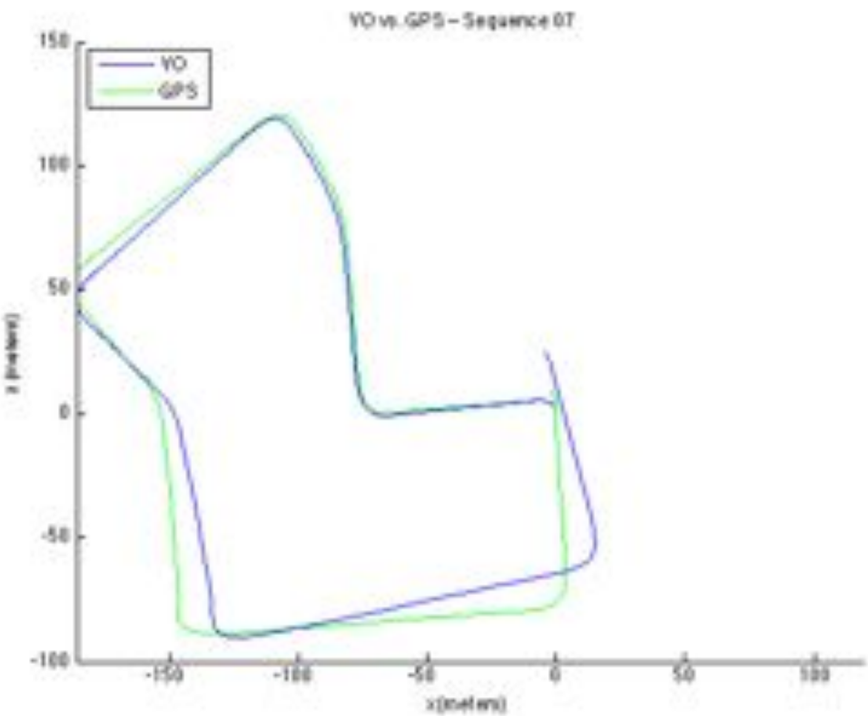


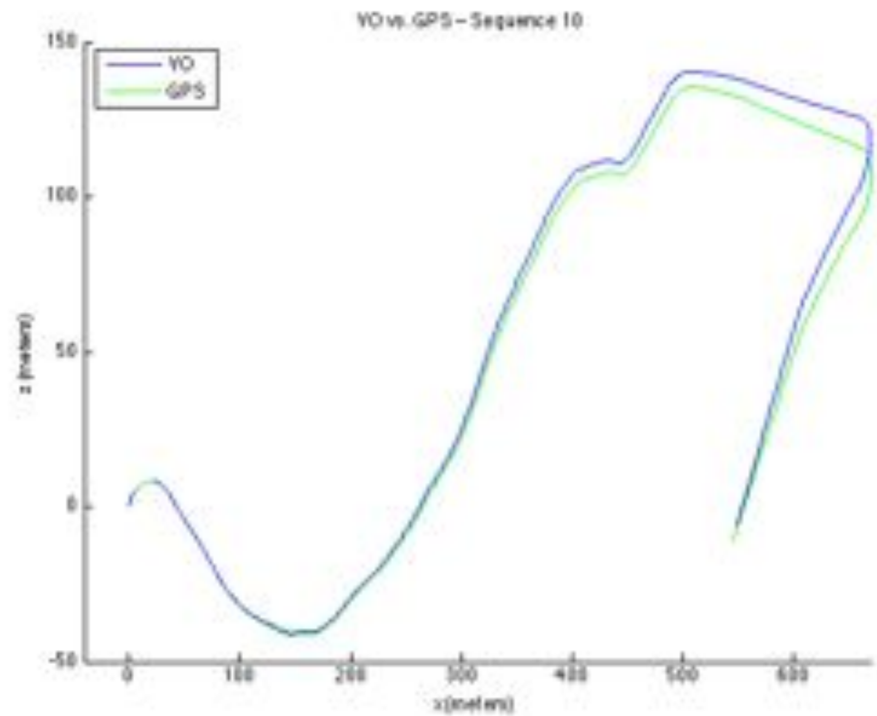


http://www.cvlibs.net/datasets/kitti/

Results on KITTI Benchmark

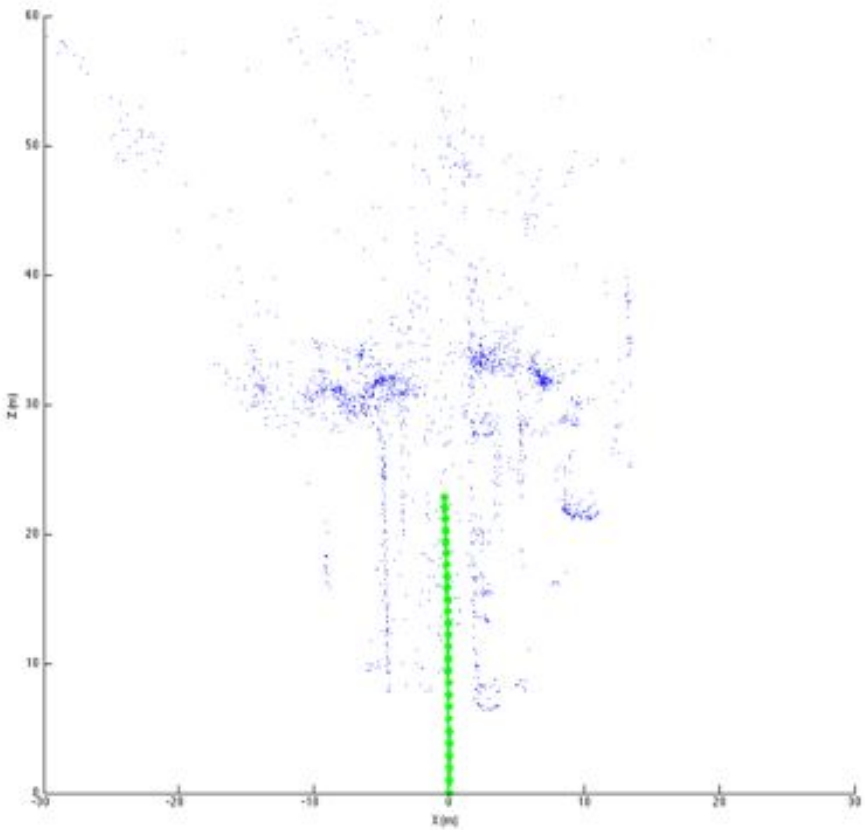
- Representative results on KITTI VO Benchmark
  - Average translational/rotational errors are very small
  - Accumulate over time, resulting in drift





GTSAM VO Example with Point Cloud

- StereoVOExample_large.m in GTSAM
  - Takes VO output and improves result through bundle adjustment (more on that later!)



tinyurl.com/gtsam