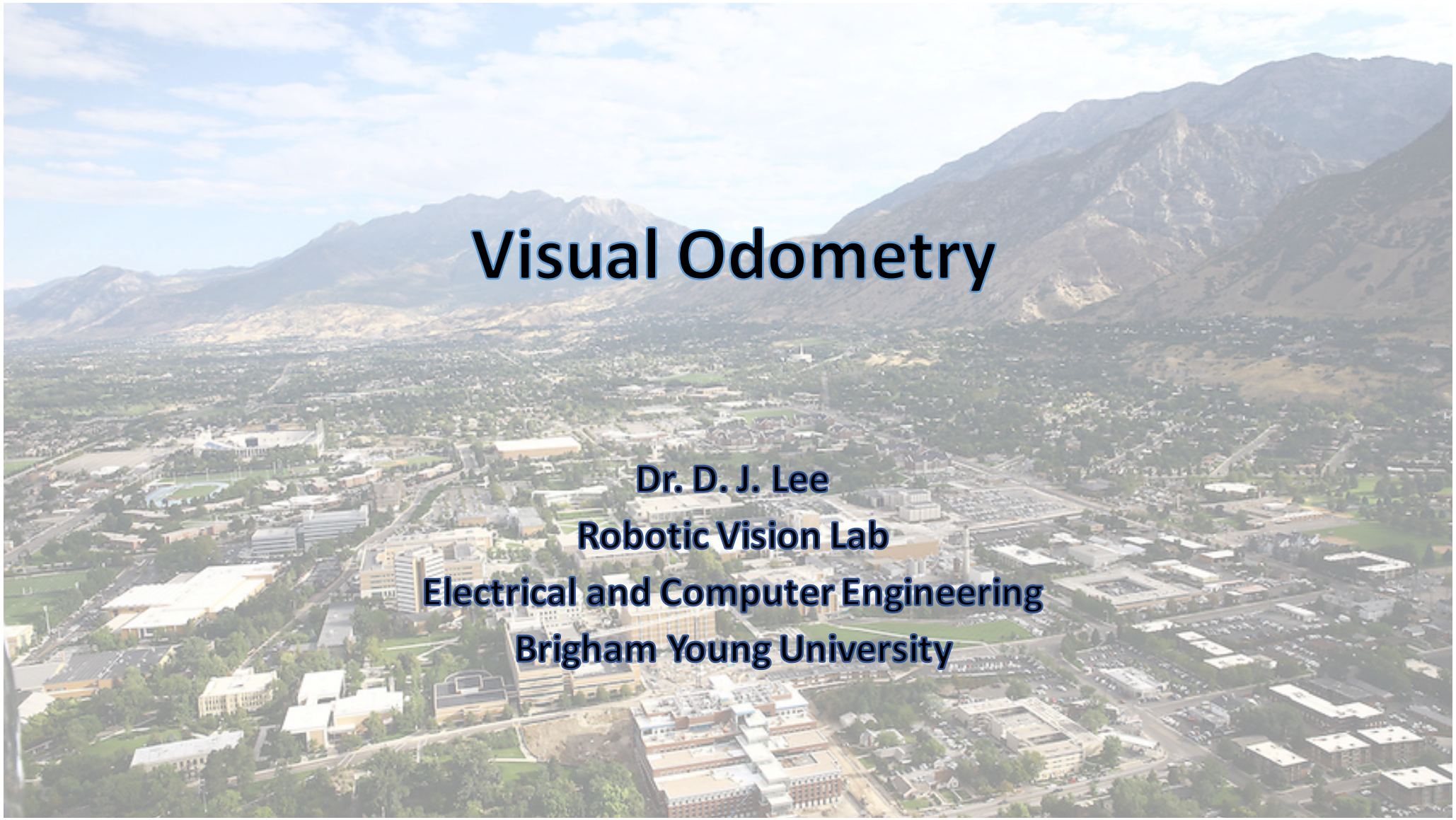


Visual Odometry

Dr. D. J. Lee
Robotic Vision Lab
Electrical and Computer Engineering
Brigham Young University

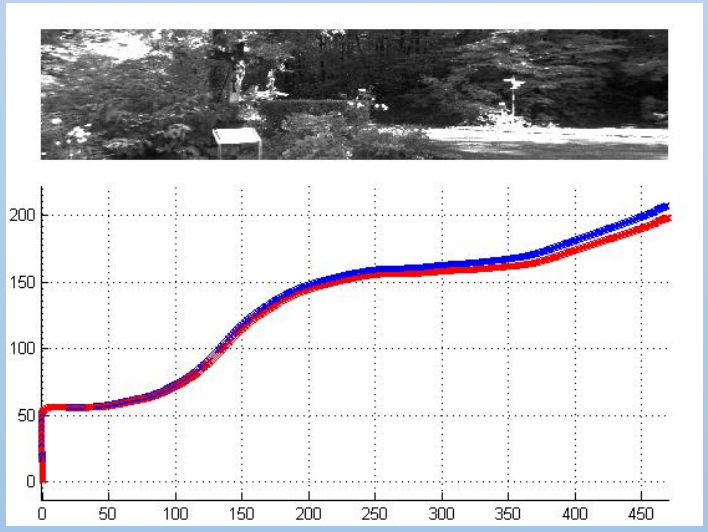
What is Visual Odometry (VO) ?

- VO is the process of incrementally estimating the pose of the vehicle by examining the changes that motion induces on the images of its onboard cameras.

Input

Image sequence (or video stream) from one or more cameras attached to a moving vehicle

Output



Estimates of R & T between two frames

What is Visual Odometry (VO) ?

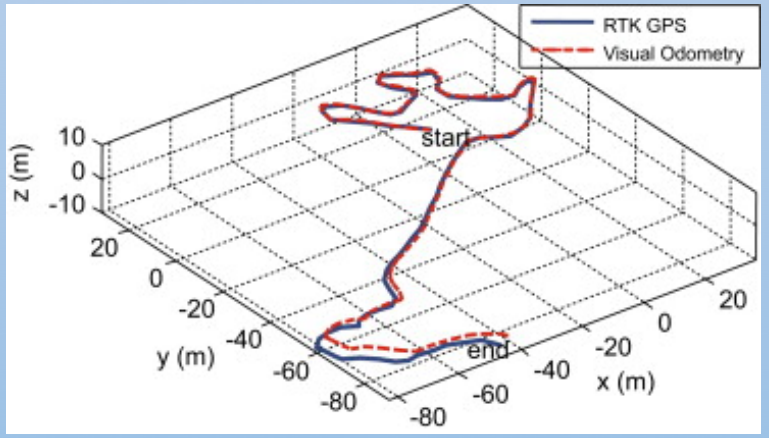
- Visual Odometry
  - Usage of visual information as a sensor
  - Realization of the real-time navigation system using 3D reconstruction algorithms (camera motion estimation algorithm)

Assumptions

- Sufficient illumination in the environment
- Dominance of static scene over moving objects
- Enough texture to allow apparent motion to be extracted
- Sufficient scene overlap between consecutive frames

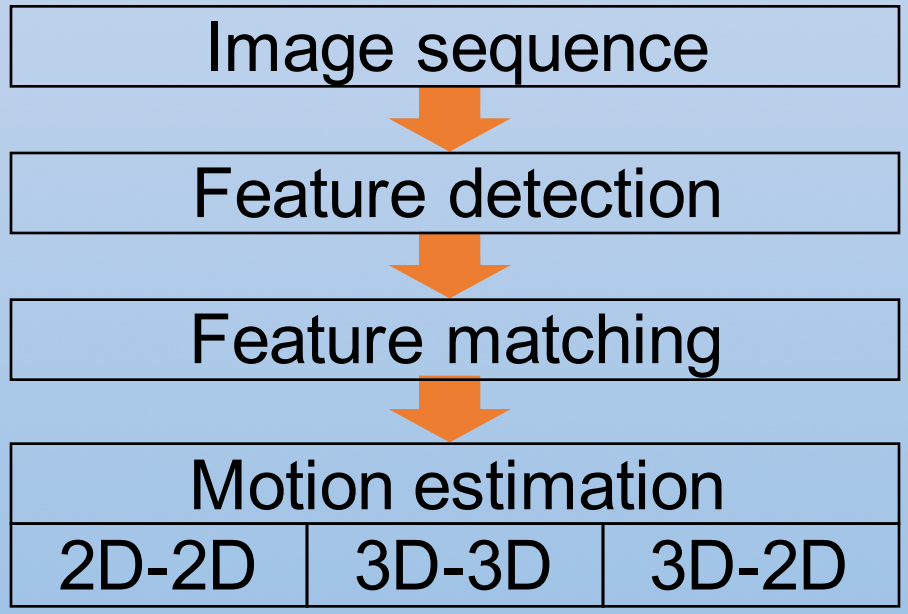
Why VO ?

- Contrary to wheel odometry, VO is not affected by wheel slip in uneven terrain or other adverse conditions.
- More accurate trajectory estimates compared to wheel odometry (relative position error 0.1% – 2%)
- VO can be used as a complement to
  - wheel odometry
  - GPS
  - inertial measurement units (IMUs)
  - laser odometry
- VO works in GPS-denied environments, such as underwater and aerial



VO Flow Chart

- VO computes the camera path incrementally (pose after pose)



2D-to-2D Algorithm

1) Capture a new frame I_k
2) Extract and match features between I_{k-1} and I_k
3) Compute the essential matrix for image pair I_{k-1}, I_k
4) Decompose the essential matrix into R_k and t_k, and form a rigid body transformation T_k
5) Compute relative scale and rescale t_k accordingly
6) Concatenate transformations to get $C_k = C_{k-1}T_k$

$$, \textit{where } T_k = \begin{bmatrix} R_k & t_k \\ 0 & 1 \end{bmatrix}$$

7) Repeat from 1)

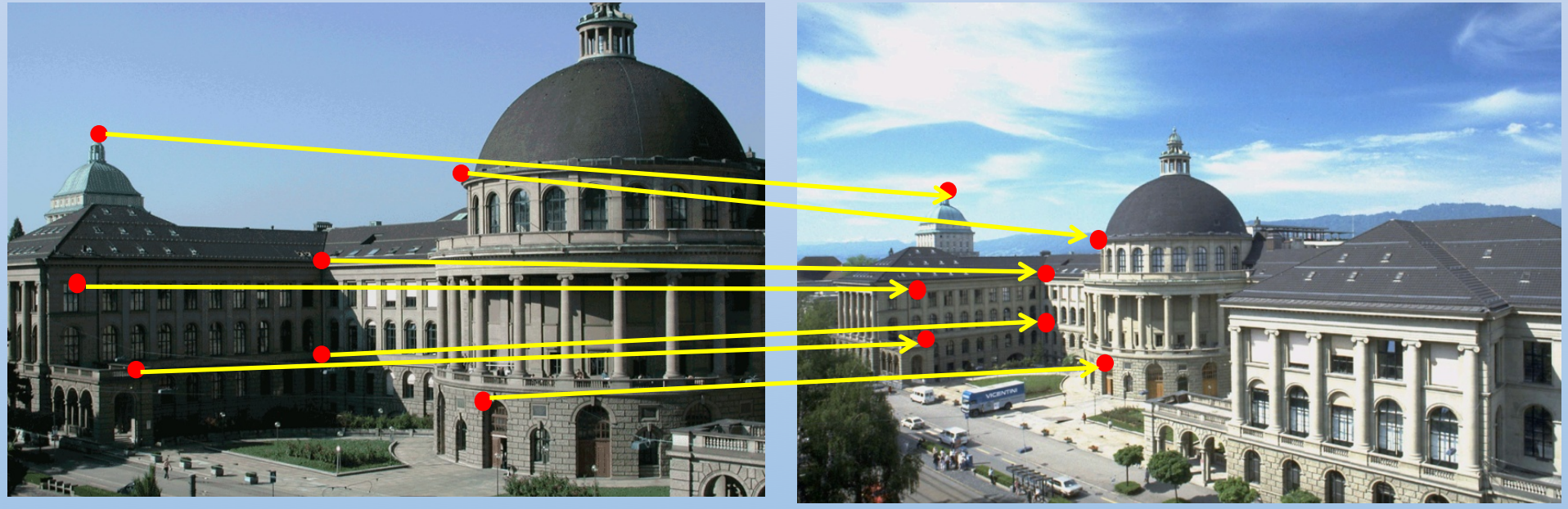
3D-to-3D Algorithm

1) Capture two stereo image pairs $I_{l\ k-1}$, $I_{r\ k-1}$ and $I_{l\ k}$, $I_{r\ k}$
2) Extract and match features between $I_{l\ k-1}$ and $I_{l\ k}$
3) Triangulate matched features for each stereo pair
4) Compute T_k from 3-D features X_{k-1} and X_k
5) Concatenate transformation by computing $C_k = C_{k-1}T_k$
6) Repeat from 1).

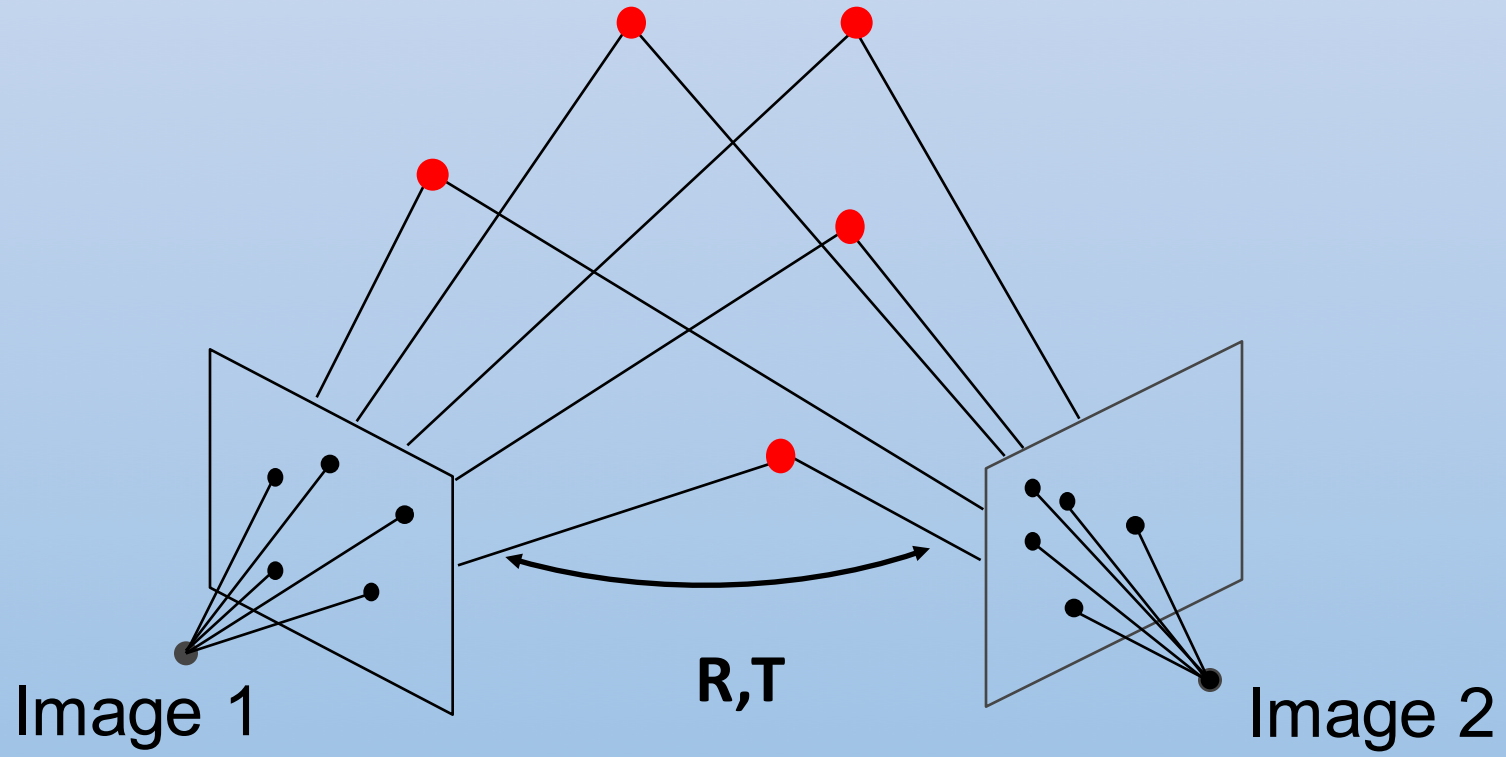
3D-to-2D Algorithm

1) Do only once:
   1.1) Capture two frames I_{k-2}, I_{k-1}
   1.2) Extract and match features between them
   1.3) Triangulate features from I_{k-2}, I_{k-1}
2) Do at each iteration:
   2.1) Capture new frame I_k
   2.2) Extract features and match with previous frame I_{k-1}
   2.3) Compute camera pose (PnP) from 3-D-to-2-D matches
   2.4) Triangulate all new feature matches between I_k and I_{k-1}
   2.5) Iterate from 2.1).

Working Principle



Working Principle



Computing the Essential Matrix

- The essential matrix can be computed from 5 point correspondences using [Nister'2003] algorithm (5-point algorithm)
- The 5-p algorithm has become the standard for 2D-2D motion estimation, however, its implementation is not straightforward.
- A simple and straightforward solution for n>=8 noncoplanar points is the Longuet-Higgins' 8-p algorithm,

$$p_r^T E p_l = \begin{bmatrix} x_r & y_r & z_r \end{bmatrix} E \begin{bmatrix} x_l \\ y_l \\ z_l \end{bmatrix} = a^T E^s = 0$$

$$a = \begin{bmatrix} x_l x_r & y_l x_r & z_l x_r & x_l y_r & y_l y_r & z_l y_r & x_l z_r & y_l z_r & z_l z_r \end{bmatrix}$$

$$E^s = \begin{bmatrix} e_{11} & e_{12} & e_{13} & e_{21} & e_{22} & e_{23} & e_{31} & e_{32} & e_{33} \end{bmatrix}^T$$

Computing the Essential Matrix

- Since the intrinsic parameters are known, OpenCV function findFundamentalMatrix() can be used to find F.
- Essential matrix E can be determined from F

$$\bar{p}_r^T F \, \bar{p}_l = 0$$

$$E = (M_r)^T F \, M_l$$

$\bar{p}_l$ and $\bar{p}_r$ are in pixel frame

If images are captured by the same camera, then $M = M_r = M_l$

- SVD of E to find four sets of R and T and pick one set that gives all positive Z.

OpenCV Function

Mat findEssentialMat(InputArray points1, InputArray points2, double focal=1.0, Point2d pp=Point2d(0, 0), int method=RANSAC, double prob=0.999, double threshold=1.0, OutputArray mask=noArray())

- **points1** – Array of N (N >= 5) 2D points from the first image. The point coordinates should be floating-point (single or double precision).
- **points2** – Array of the second image points of the same size and format as points1 .
- **focal** – focal length of the camera. Note that this function assumes that points1 and points2 are feature points from cameras with same focal length and principle point.
- **pp** – principle point of the camera (C_x and C_y of the intrinsic parameters)
- **method** –Method for computing a fundamental matrix. RANSAC for the RANSAC algorithm. MEDS for the LMedS algorithm.
- **threshold** – Parameter used for RANSAC. It is the maximum distance from a point to an epipolar line in pixels, beyond which the point is considered an outlier and is not used for computing the final fundamental matrix. It can be set to something like 1-3, depending on the accuracy of the point localization, image resolution, and the image noise.
- **prob** – Parameter used for the RANSAC or LMedS methods only. It specifies a desirable level of confidence (probability) that the estimated matrix is correct.
- **mask** – Output array of N elements, every element of which is set to 0 for outliers and to 1 for the other points. The array is computed only in the RANSAC and LMedS methods.

findEssentialMat

- This function estimates essential matrix based on the five-point algorithm solver in [Nister03]. The epipolar geometry is described by the following equation:

$$[p_2; 1]^T K^T E K [p_1; 1] = 0$$

$$K = \begin{bmatrix} f & 0 & x_{pp} \\ 0 & f & y_{pp} \\ 0 & 0 & 1 \end{bmatrix}$$

- where E is an essential matrix, p_1 and p_2 are corresponding points in the left and the right images, respectively. The result of this function may be passed further to decomposeEssentialMat() or recoverPose() to recover the relative pose between cameras.

Solving R and T from E

- Essential Matrix

$$E = \hat{T}R \in R^{3\times 3}$$

- A nonzero matrix E is an essential matrix if and only if E has a singular value decomposition (SVD)$E=UDV^T$ with

$$D=\mathrm{diag}\{\sigma, \sigma, 0\}$$

- *There exist exactly four solutions (2 with positive Z)*

$$\text{If a given matrix } E \in R^{3\times 3} \text{ has SVD } E = UDV^T \text{ then}$$

$$(\hat{T}_1, R_1) = (UR_z(+\frac{\pi}{2})DU^T, UR_z^T(+\frac{\pi}{2})V^T)$$

$$(\hat{T}_2, R_2) = (UR_z(-\frac{\pi}{2})DU^T, UR_z^T(-\frac{\pi}{2})V^T)$$

R_z: Rotation matrix about Z axis

Estomate R and T

Essential Matrix $E = \hat{T}R \in R^{3\times 3}$

Compute SVD of the new E=UDV^T

Calculate R and T as

$$R_Z^T(\pm\frac{\pi}{2}) = \begin{bmatrix} 0 & \pm 1 & 0 \\ \mp 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R = UR_z^T(\pm\frac{\pi}{2})V^T$$

$$\Sigma = \begin{bmatrix} \sigma & 0 & 0 \\ 0 & \sigma & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\hat{T} = UR_z(\pm\frac{\pi}{2})\Sigma U^T$$

$$\hat{T} = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix}$$

Normalized the unknown scale factor can be calculated if object size is known.

The solution is up to an unknown scale factor if T is not given.
T can be obtained from IMU, GPS, or wheel encoder.

decomposeEssentialMat

void decomposeEssentialMat(InputArray **E**, OutputArray **R1**, OutputArray **R2**, OutputArray **t**)

This function decompose E using svd decomposition [HartleyZ00].

There are 4 possible poses exists for a given E. They are $[R_2, -t]$, $[R_2, t]$, $[R_1, -t]$, $[R_1, t]$.

By decomposing E, you can only get the direction of the translation, so the function returns unit t.

- **E** – The input essential matrix.
- **R1** – One possible rotation matrix.
- **R2** – Another possible rotation matrix.
- **t** – One possible translation.

recoverPose

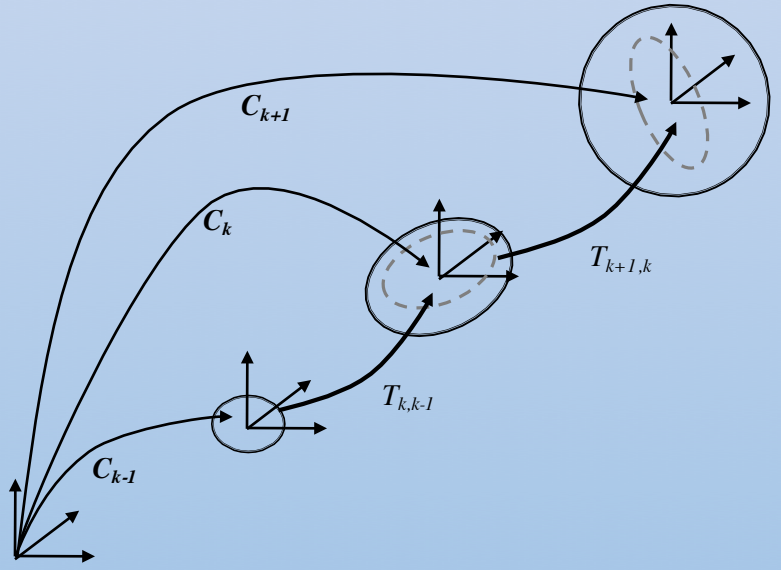
int recoverPose(InputArray E, InputArray points1, InputArray points2, OutputArray R, OutputArray t, doublefocal=1.0, Point2d pp=Point2d(0, 0), InputOutputArray mask=noArray())

This function decompose E using decomposeEssentialMat() and then then verifies possible pose hypotheses to return one good R and t.

- **E** – The input essential matrix.
- **points1** – Array of N 2D points from the first image. The point coordinates should be floating-point (single or double precision).
- **points2** – Array of the second image points of the same size and format as points1 .
- **R** – Recovered relative rotation.
- **t** – Recoverd relative translation.
- **focal** – Focal length of the camera. Note that this function assumes that points1 and points2 are feature points from cameras with same focal length and principle point.
- **pp** – Principle point of the camera.
- **mask** – Input/output mask for inliers in points1 and points2. If it is not empty, then it marks inliers in points1 and points2 for then given essential matrix E. Only these inliers will be used to recover pose. In the output mask only inliers which pass the cheirality check.

VO Drift

- The errors introduced by each new frame-to-frame motion accumulate over time.
- This generates a drift of the estimated trajectory from the real path.
- The uncertainty of the camera pose at C_k is a combination of the uncertainty at C_{k-1} (black solid ellipse) and the uncertainty of the transformation $T_{k,k-1}$ (gray dashed ellipse).



- Camera motion between time k -1 and k is rearranged in the form of the rigid body transformation $T_k \in R^{4x4}$:

$$T_k = \begin{bmatrix} R_k & t_k \\ 0 & 1 \end{bmatrix}$$

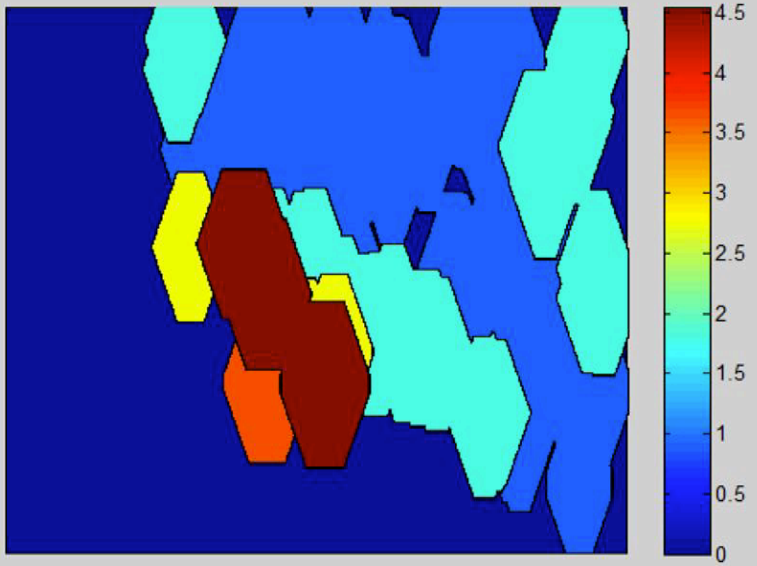
, where $R_k \in R^{3x3}$ is the rotation matrix and $t_k \in R^{3x1}$ is the translation vector.

Camera pose $C_k = C_{k-1} \cdot T_k$

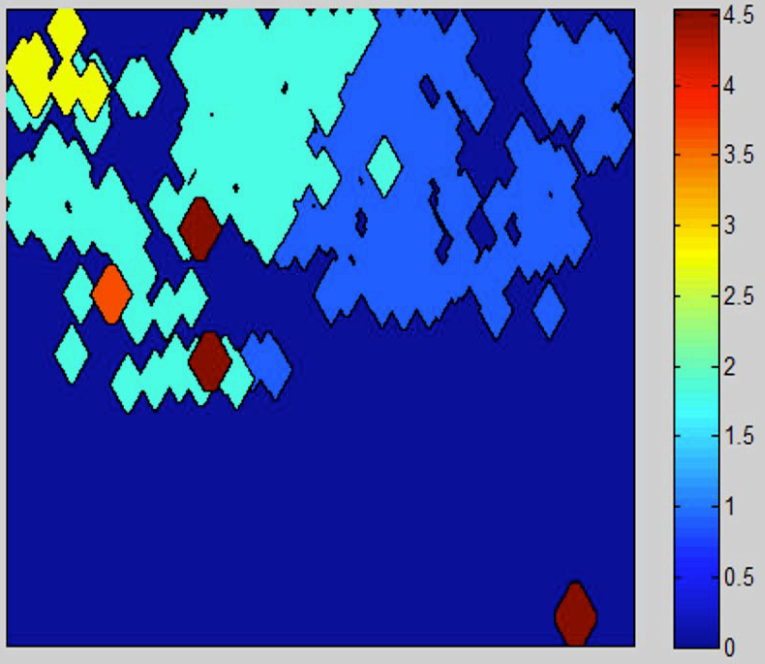
Motion Estimation: Summary

Type of Correspondences	Monocular	Stereo
2D-2D	X	X
3D-3D		X
3D-2D	X	X

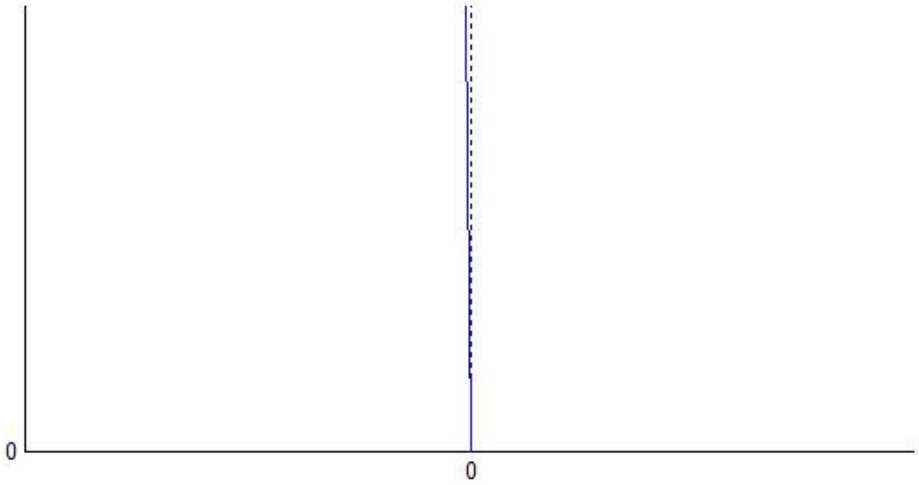
Motion Vector Analysis

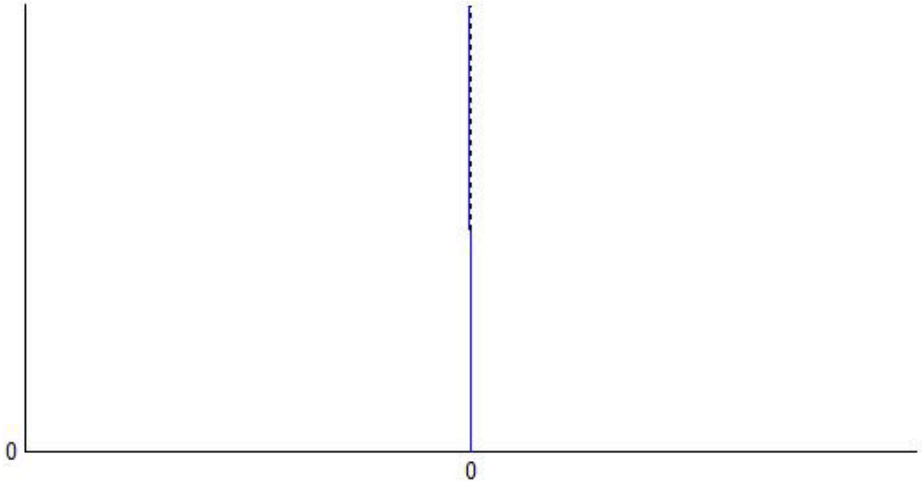


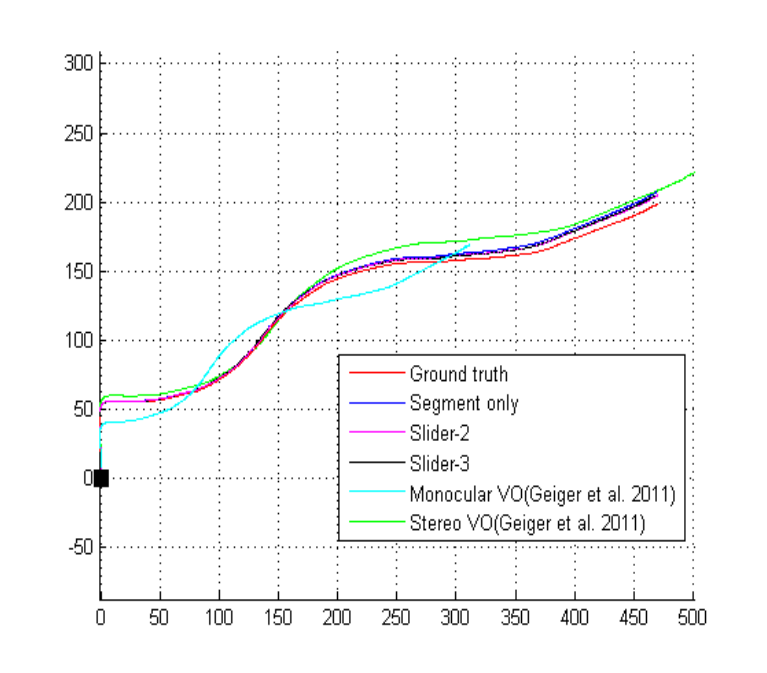
Motion Vector Analysis

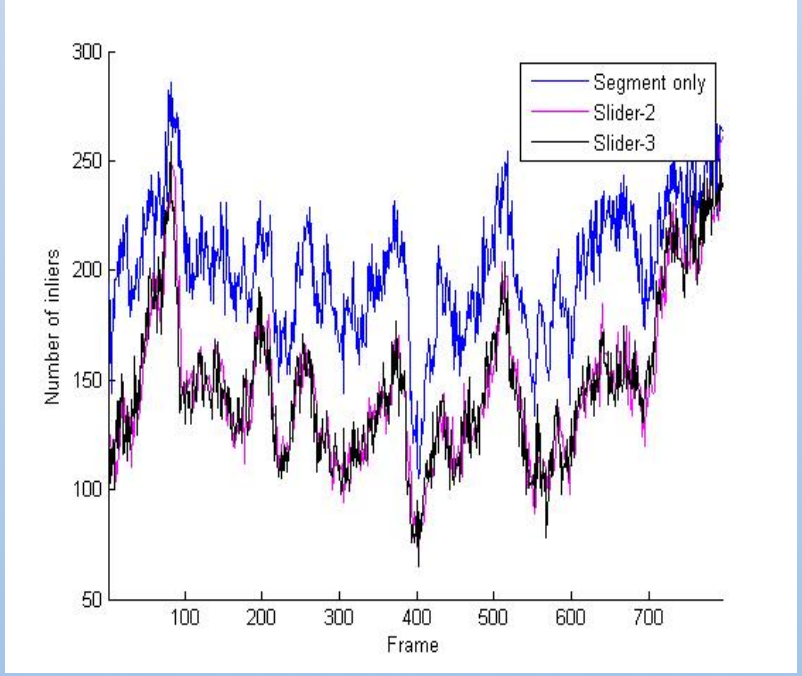


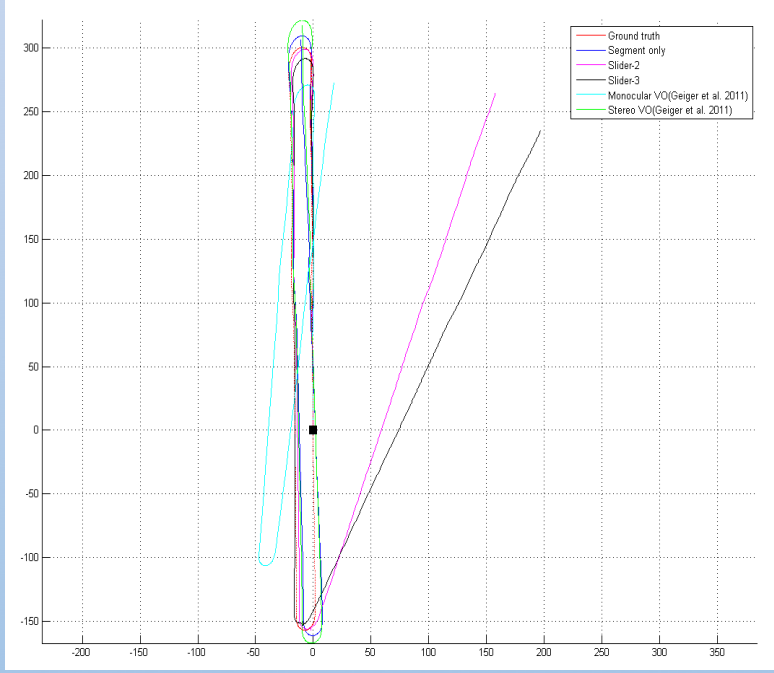
Results

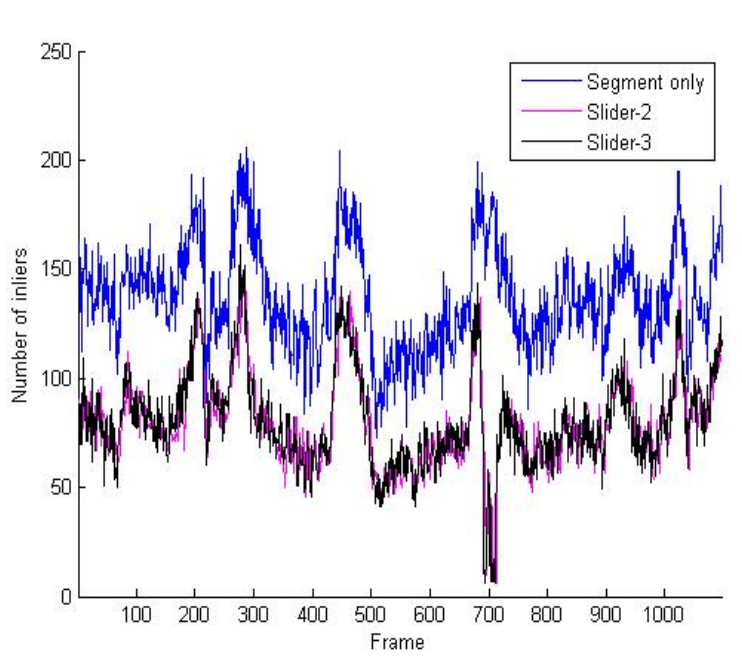




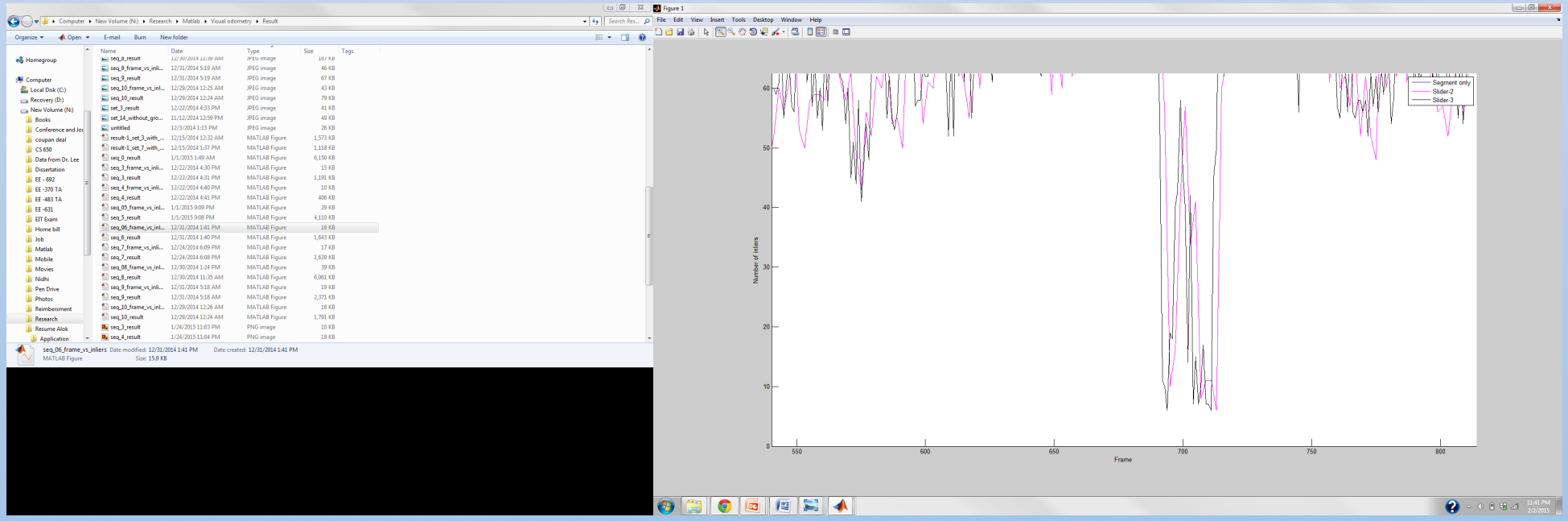

300
250
200
150
100
50
0
-50
0
50
100
150
200
250
300
350
400
450
500
Ground truth
Segment only
Slider-2
Slider-3
Monocular VO(Geiger et al. 2011)
Stereo VO(Geiger et al. 2011)

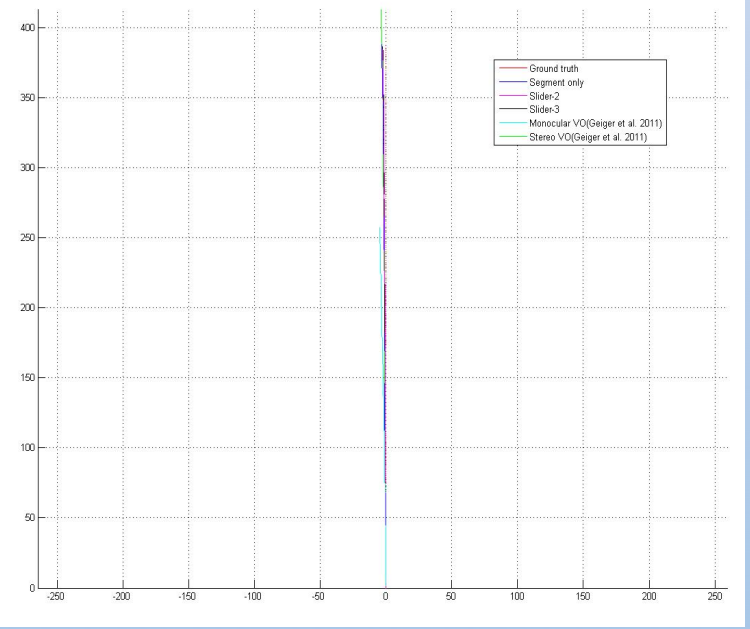

300
250
200
150
100
50
Number of inliers
100
200
300
400
500
600
700
Frame
Segment only
Slider-2
Slider-3

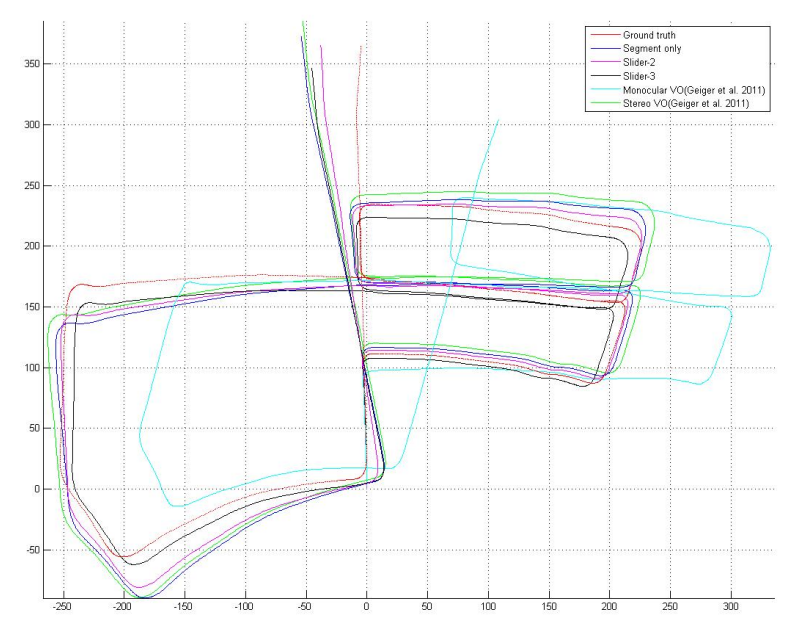

Ground truth
Segment only
Slider-2
Slider-3
Monocular VO(Geiger et al. 2011)
Stereo VO(Geiger et al. 2011)

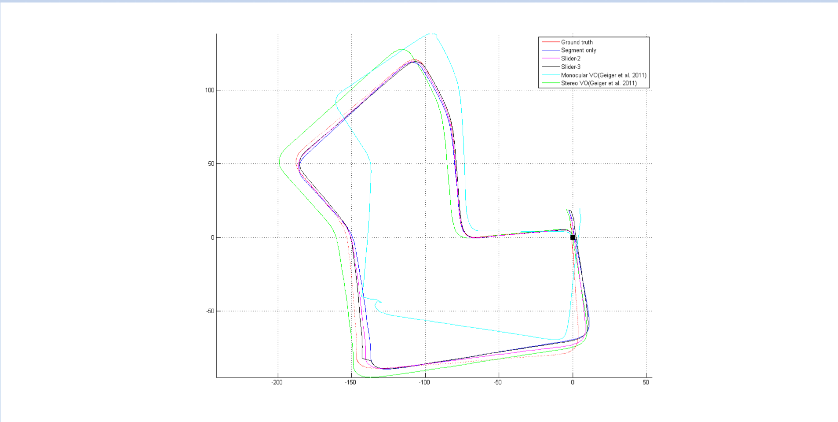

Segment only
Slider-2
Slider-3
Number of inliers
Frame

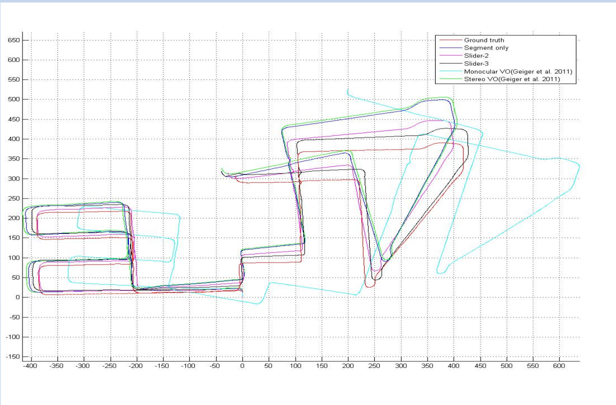
Correspondences are not sufficient

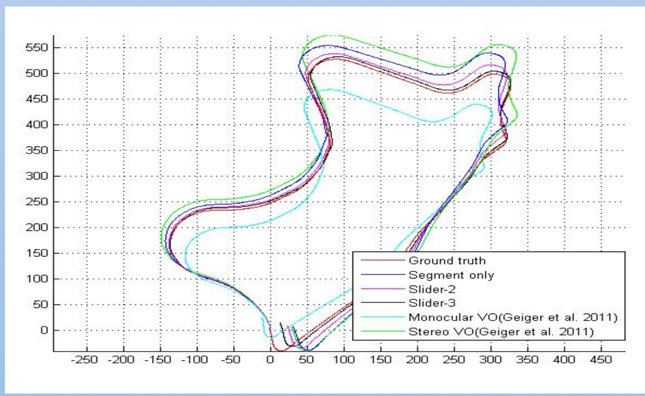


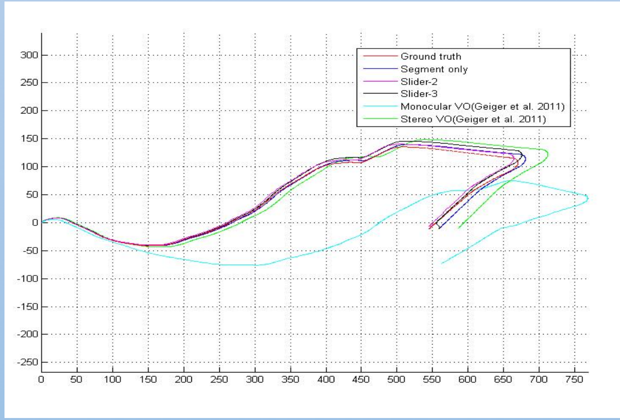

Ground truth
Segment only
Slider-2
Slider-3
Monocular VO(Geiger et al. 2011)
Stereo VO(Geiger et al. 2011)


Ground truth
Segment only
Slider-2
Slider-3
Monocular VO(Geiger et al. 2011)
Stereo VO(Geiger et al. 2011)


Ground truth
Segment only
Slider-2
Slider-3
Monocular VO(Geiger et al. 2011)
Stereo VO(Geiger et al. 2011)


Ground truth
Segment only
Slider-2
Slider-3
Monocular VO(Geiger et al. 2011)
Stereo VO(Geiger et al. 2011)


Ground truth
Segment only
Slider-2
Slider-3
Monocular VO(Geiger et al. 2011)
Stereo VO(Geiger et al. 2011)


Ground truth
Segment only
Slider-2
Slider-3
Monocular VO(Geiger et al. 2011)
Stereo VO(Geiger et al. 2011)

Relative Error

Seq.	Configuration		Relative Error (%)				
	Distance	Environment	Segment	Slider - 2	Slider - 3	Mono -VO	Stereo-VO
3	558.67m	Country	0.3947	0.2830	0.1811	44.7308	6.7982
4	385.46m	Country	0.6873	0.0645	0.0982	49.6110	6.3451
5	2199.14m	Urban	3.9923	1.9180	3.1245	0.2992	6.5651
6	1228.59m	Urban	2.9848	0.4273	2.9436	17.3283	6.4599
7	694.37m	Urban	0.4822	0.3845	0.1918	3.4431	6.0926
8	3209.63m	Urban+ Country	5.0965	0.2698	4.1638	6.8845	6.0486
9	1698.84m	Urban+ Country	2.7115	0.1013	0.6596	13.0740	6.2188
10	917.34m	Urban+ Country	1.5221	0.7241	2.2806	14.4436	6.0152

KITTI Dataset - 3. Comparison between Segment, Slider-2, Slider-3, Mono - VO and Stereo-VO

	MSE Error				RMSE Error			
	Rotation			Translation	Rotation			Translation
Segment	0.0000	0.0000	0.0000	0.0001	0.0000	0.0024	0.0011	0.0095
	0.0000	0.0000	0.0000	0.0001	0.0024	0.0000	0.0025	0.0080
	0.0000	0.0000	0.0000	0.0013	0.0011	0.0025	0.0000	0.0358
Slider-2	0.0000	0.0000	0.0000	0.0001	0.0000	0.0005	0.0002	0.0119
	0.0000	0.0000	0.0000	0.0001	0.0005	0.0000	0.0005	0.0096
	0.0000	0.0000	0.0000	0.0002	0.0002	0.0005	0.0000	0.0133
Slider-3	0.0000	0.0000	0.0000	0.0003	0.0000	0.0007	0.0003	0.0169
	0.0000	0.0000	0.0000	0.0002	0.0007	0.0000	0.0006	0.0135
	0.0000	0.0000	0.0000	0.0003	0.0003	0.0006	0.0000	0.0178
Mono VO	0.0036	0.0004	0.0108	1461.4365	0.0598	0.0203	0.1040	38.2287
	0.0001	0.0000	0.0008	17.2377	0.0116	0.0038	0.0288	4.1518
	0.0111	0.0006	0.0040	438.4018	0.1053	0.0236	0.0632	20.9380
Stereo VO	0.0000	0.0000	0.0000	0.0001	0.0000	0.0023	0.0011	0.0097
	0.0000	0.0000	0.0000	0.0001	0.0023	0.0000	0.0025	0.0083
	0.0000	0.0000	0.0000	0.0027	0.0011	0.0025	0.0000	0.0521

Improving the Accuracy of VO

- Other sensors can be used such as
  - IMU (called inertial VO)
  - Compass
  - GPS
  - Laser
- An IMU combined with a single camera allows the estimation of the absolute scale.