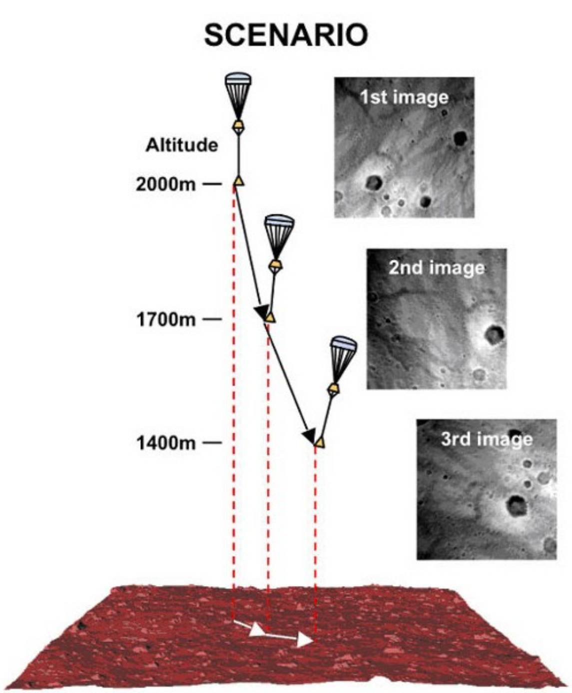
Visual Odometry

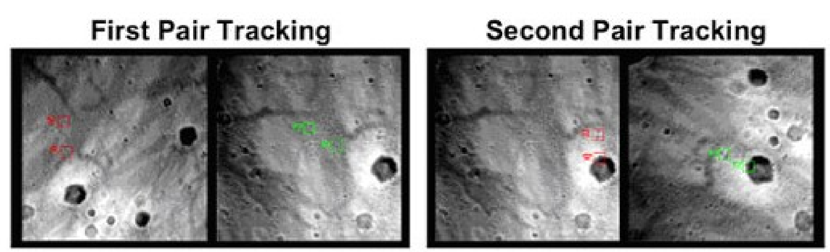
Features, Tracking, Essential Matrix, and RANSAC

Jet Propulsion Laboratory
California Institute of Technology

Stephan Weiss
Computer Vision Group
NASA-JPL / CalTech

Stephan.Weiss@ieee.org

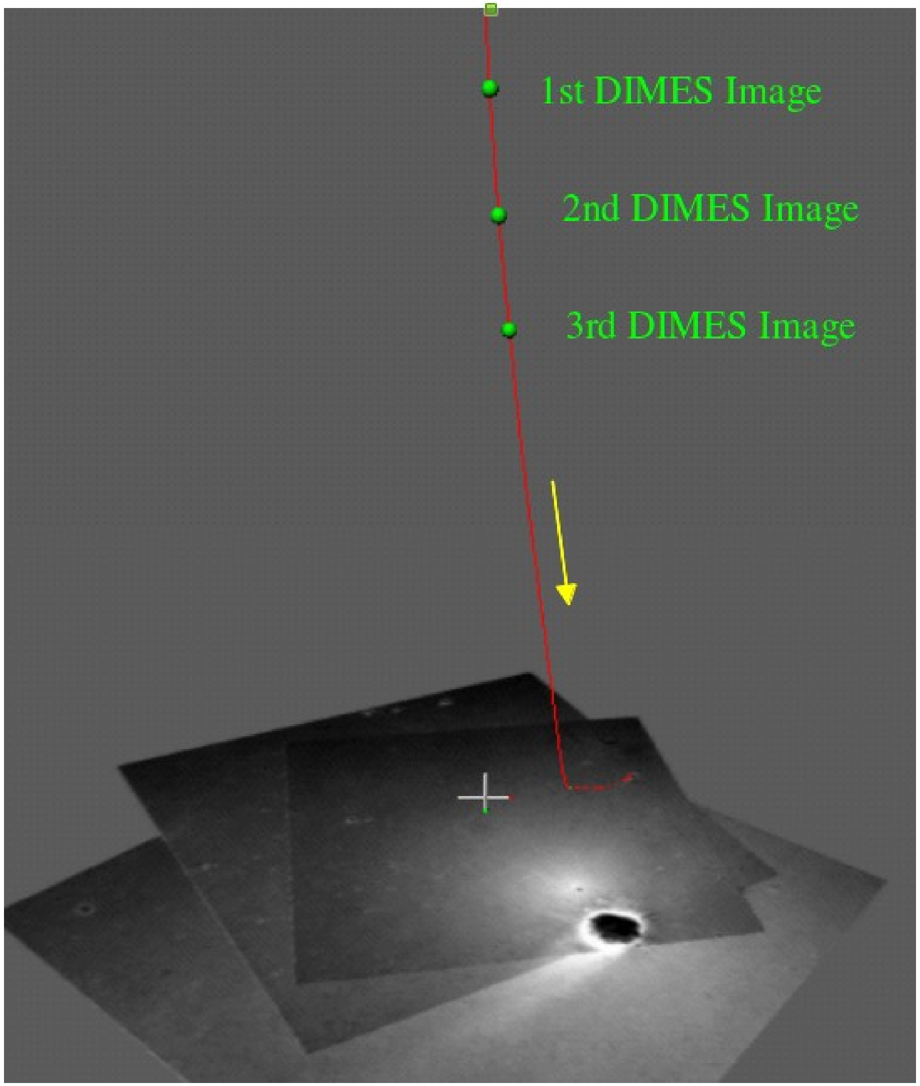




MER-A/Spirit, Gusev Crater, January 4th, 2004

Outline

- The Camera as a sensor
- Camera motion estimation: the essential matrix
- Dealing with noise: RANSAC
- Getting to the point
- Keeping the point

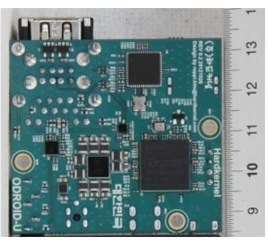


Opportunity EDL Trajectory

Camera Motion Estimation

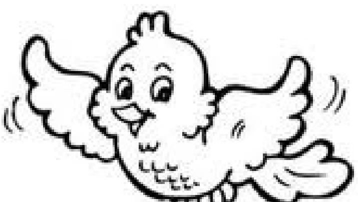
- Why using a camera?
  - Vast information
  - Extremely low **S**ize, **W**eight, **a**nd **P**ower (SWaP) footprint
  - Cheap and easy to use
  - Passive sensor
  - Processing power is OK today

Cellphone type camera, up to 16Mp (480MB/s @ 30Hz)

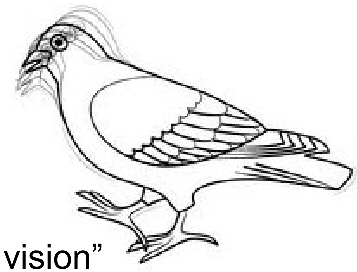


Cellphone processor unit
1.7GHz quadcore ARM <10g

After all, it's what nature uses, too!



“stereo vision”

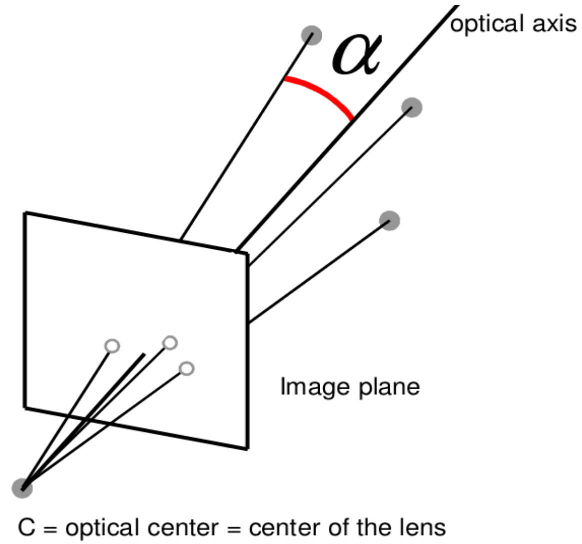


“monocular vision”

- Camera motion estimation
  - Understand the camera as a sensor
  - What information in the image is *particularly* useful
  - Estimate camera 6(5)DoF using 2 images: Visual Odometry (VO)

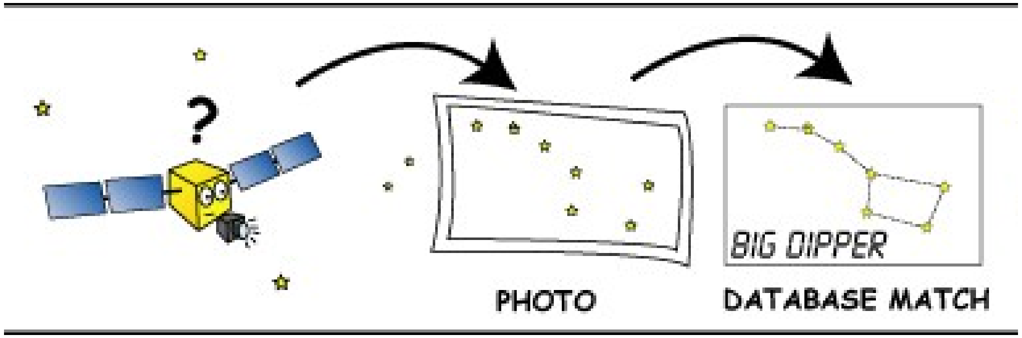
A Camera is a Bearing Sensor

- Projective sensor which measures the bearing of a point with respect to the optical axis
  - Depth can be inferred by re-observing a point from different angles
  - The movement (i.e. the *angle* between the observations) is the point's parallax

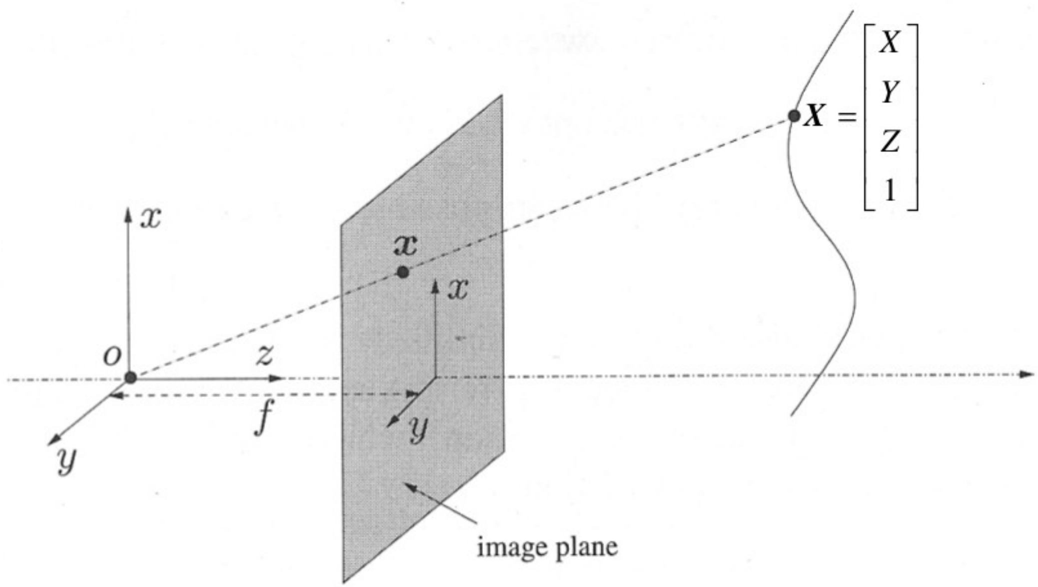


- A point at infinity is a feature which exhibits no parallax during camera motion
  - The distance of a star cannot be inferred by moving a few kilometers
  - **BUT:** it is a perfect bearing reference for attitude estimation: NASA's star tracker sensors better than 1 arc second or 0.00027deg


star tracker



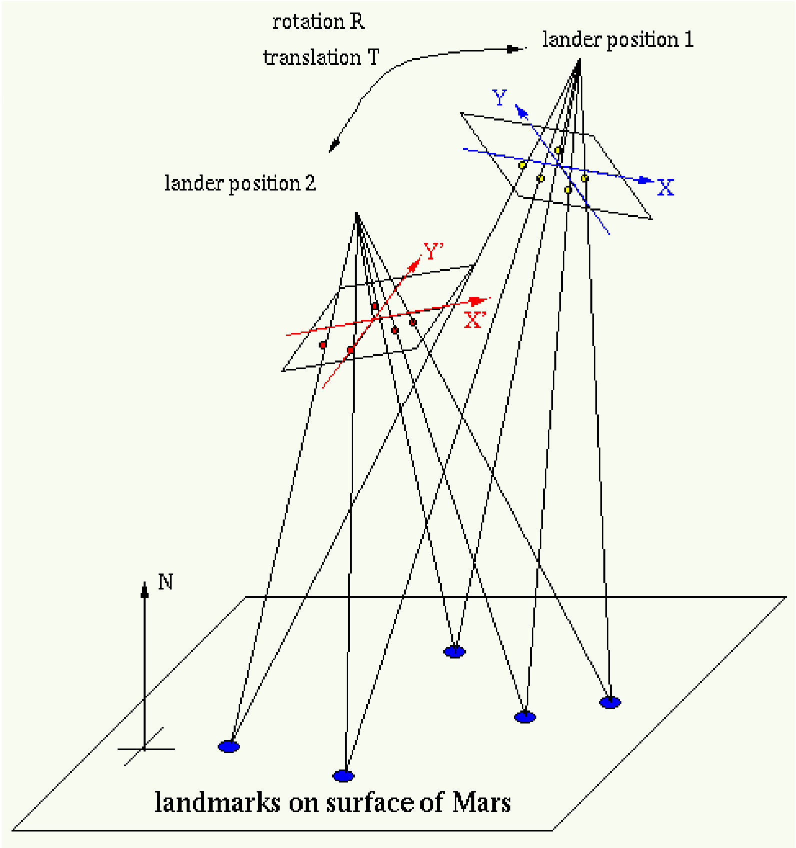
Perspective Camera – Projection on the Image Plane



Theorem of intersecting lines:

$$x = f\frac{X}{Z}, \quad y = f\frac{Y}{Z} \quad \text{or} \quad \boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix} = \frac{f}{Z}\begin{bmatrix} X \\ Y \end{bmatrix}$$

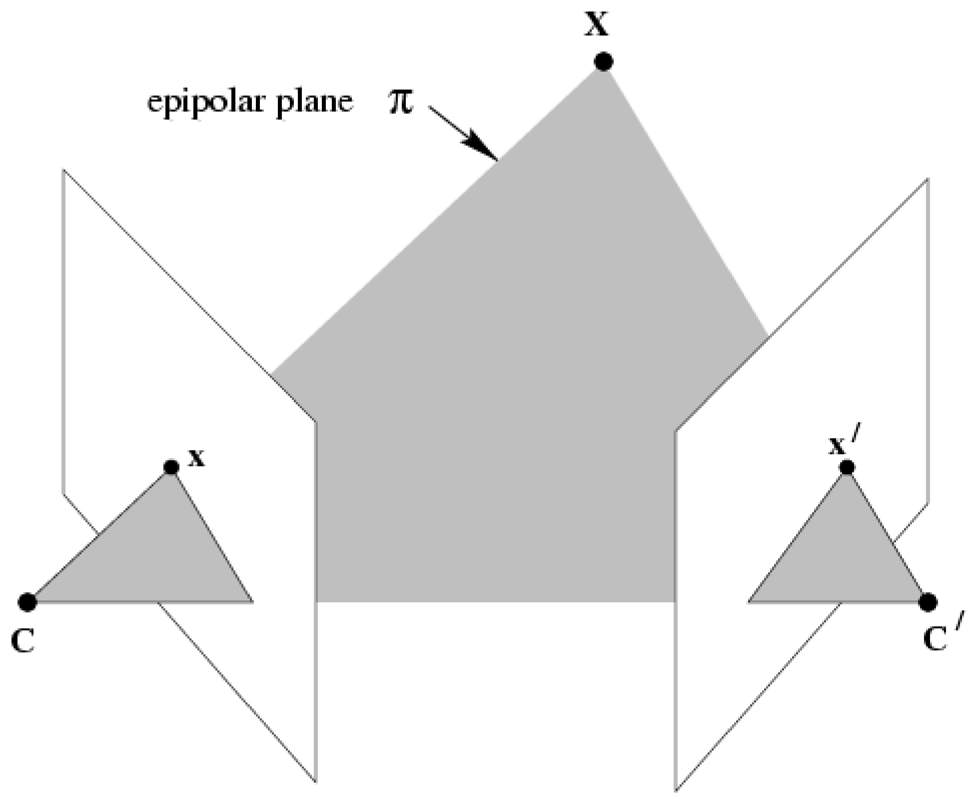
Image: Ma, Y., Soatto, S., Kosecká, J., Sastry, S.S. : "An Invitation to 3D Vision"



assume calibrated camera

Epipolar Constraint

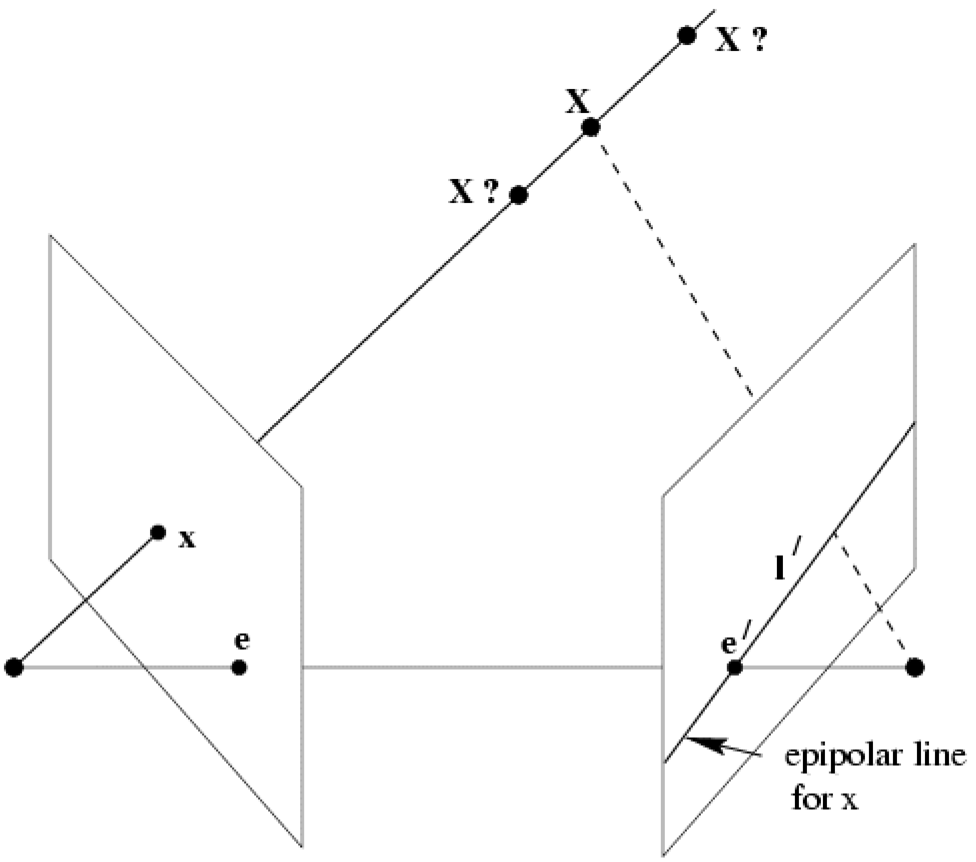
Suppose a camera undergoes motion



C,C',x,x' and X are coplanar

Epipolar Constraint

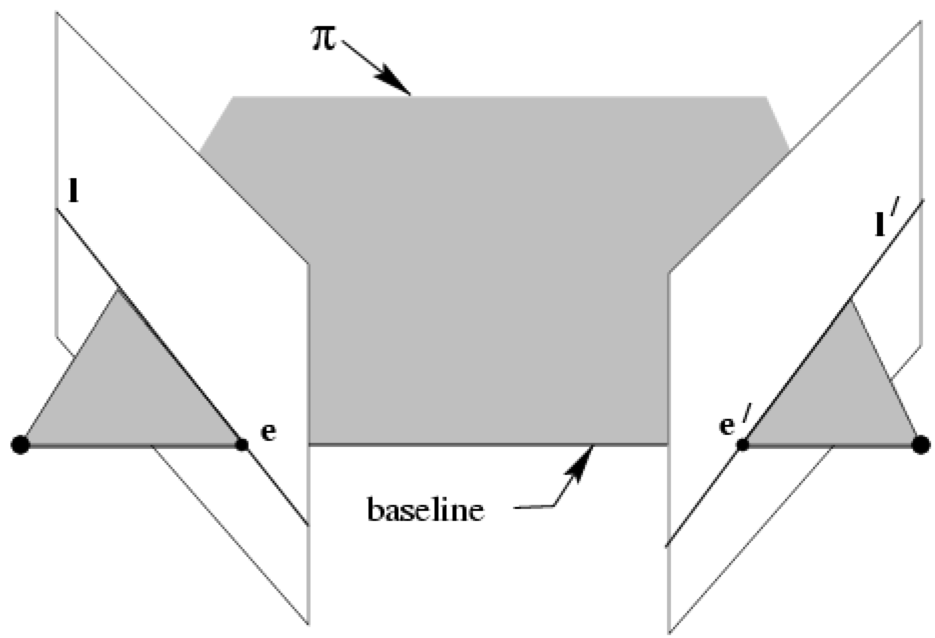
Suppose a camera undergoes motion



What if only C,C',x are known?

Epipolar Constraint

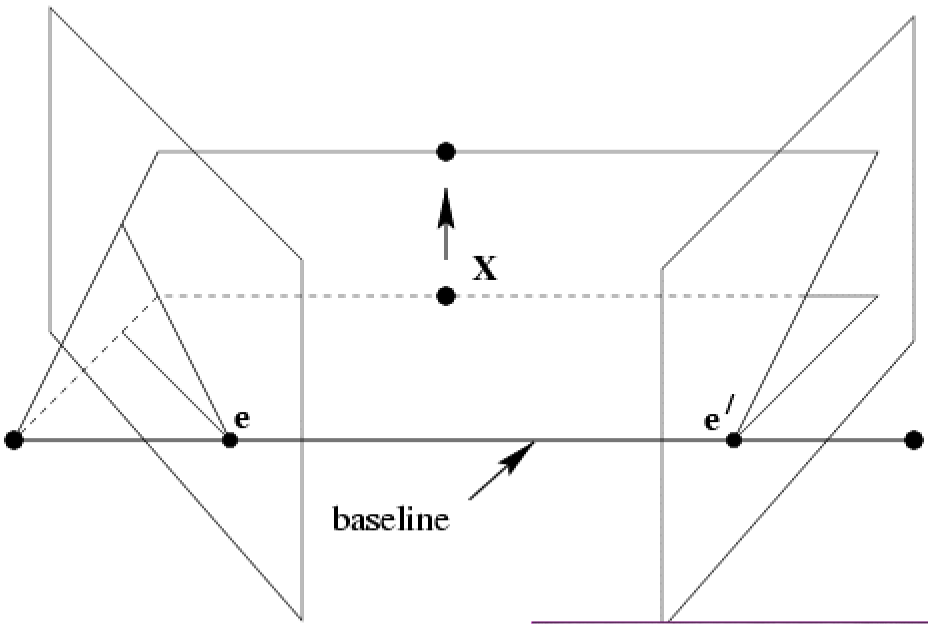
Suppose a camera undergoes motion



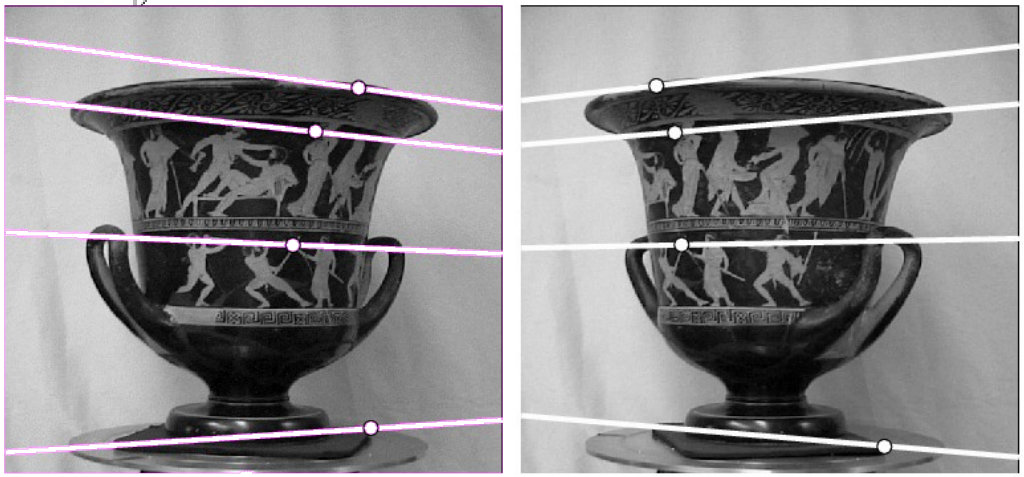
All points on π project on l and l'

Epipolar Constraint

Suppose a camera undergoes motion



Family of planes π and lines l and l'
Intersection in e and e'



Epipolar Constraint

- Formulating the epipolar constraint:

3D point transformation: $\mathbf{X}_2 = R\mathbf{X}_1 + T$

Using projected points in the image plane: $\lambda_2 \mathbf{x}_2 = R\lambda_1 \mathbf{x}_1 + T$

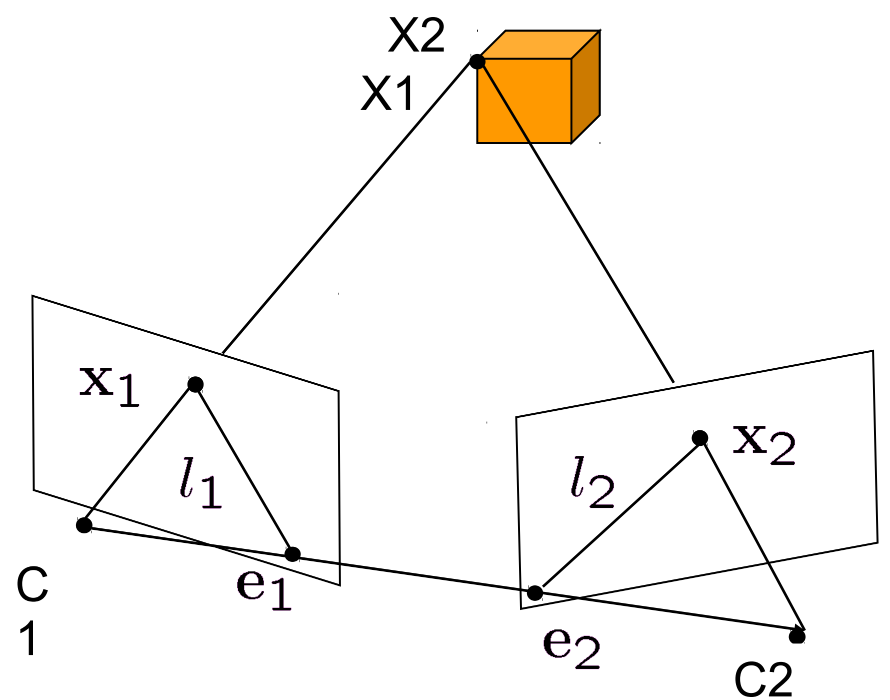
Divide by λ_1, multiply by T_X : $\lambda_2 T \mathbf{x}_2 = T R \lambda_1 \mathbf{x}_1$

Multiply by $\mathbf{x}_2$:

$$\boxed{\mathbf{x}_2^T E \mathbf{x}_1 = 0}$$

Essential Matrix:

$$E = T_{\times} R$$



Motion Estimation: Solving the Essential Matrix

$$x_2^T \mathrm{Ex} = 0$$

$$x_2 x_1 e_{11} + x_2 y_2 e_{12} + x_2 e_{13} + y_1 x_2 e_{21} + y_1 y_2 e_{22} + y_1 e_{23} + x_2 e_{31} + y_2 e_{32} + e_{33} = 0$$

separate known from unknown

$$[x_1 x_2, x_1 y_2, x_1, y_1 x_2, y_1 y_2, y_1, x_2, y_2, 1][e_{11}, e_{12}, e_{13}, e_{21}, e_{22}, e_{23}, e_{31}, e_{32}, e_{33}]^T = 0$$

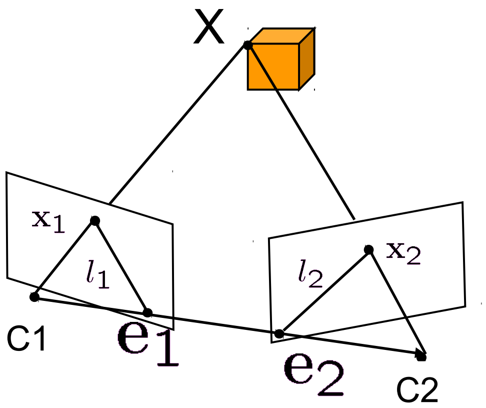
(data) (unknowns) (linear)

$$\begin{bmatrix} x_{11} x_{21} & x_{11} y_{21} & x_{11} & y_{11} x_{21} & y_{11} y_{21} & y_{11} & x_{21} & y_{21} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{1n} x_{2n} & x_{1n} y_{2n} & x_{1n} & y_{1n} x_{2n} & y_{1n} y_{2n} & y_{1n} & x_{2n} & y_{2n} & 1 \end{bmatrix} e = 0$$

$$\mathrm{Ae} = 0$$

Motion Estimation: Solving the Essential Matrix

Jet Propulsion Laboratory
California Institute of Technology



$$e_1^T E = 0 \qquad E e_2 = 0 \qquad \det E = 0 \qquad \text{rank } E = 2$$

SVD from linearly computed E matrix (rank 3)

$$E = U \begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \sigma_3 \end{bmatrix} V^T = U_1 \sigma_1 V_1^T + U_2 \sigma_2 V_2^T + U_3 \sigma_3 V_3^T$$

Compute closest rank-2 approximation $\min \|\text{E-E'}\|_E$

$$\text{E}' = U \begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & 0 \end{bmatrix} V^T = U_1 \sigma_1 V_1^T + U_2 \sigma_2 V_2^T$$

E is essential matrix if and only if
two singularvalues are equal (and third=0)

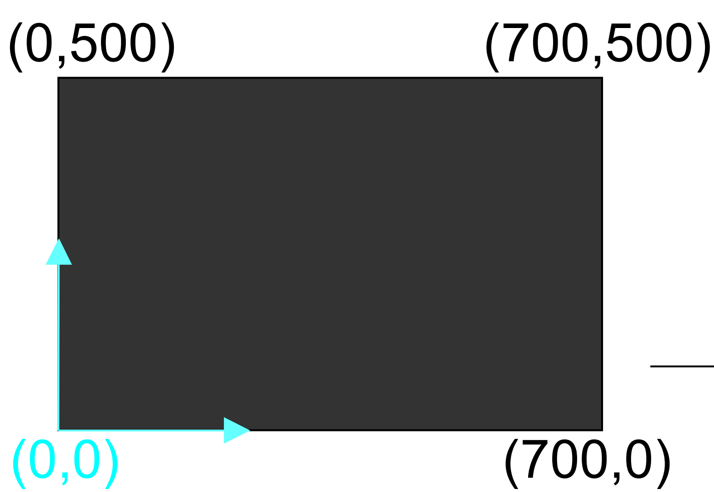
$$\text{E} = \text{Udiag}(1,1,0)\text{V}^\text{T}$$

Motion Estimation: linear 8-point algorithm

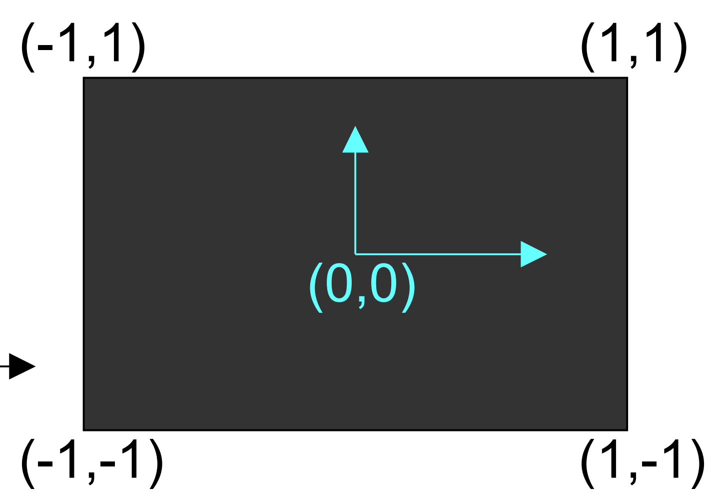
$$\begin{array}{ccccccccc} \left[\begin{matrix} x_{11}x_{21} & x_{11}y_{21} & x_{11} & y_{11}x_{21} & y_{11}y_{21} & y_{11} & x_{21} & y_{21} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{1n}x_{2n} & x_{1n}y_{2n} & x_{1n} & y_{1n}x_{2n} & y_{1n}y_{2n} & y_{1n} & x_{2n} & y_{2n} & 1 \end{matrix}\right] e = 0 \\ \sim 10000 \quad \sim 10000 \quad \sim 100 \quad \sim 10000 \quad \sim 10000 \quad \sim 100 \quad \sim 100 \quad \sim 100 \quad 1 \end{array}$$

Orders of magnitude difference
Between column of data matrix
$\rightarrow$ not normalized least-squares yields poor results

Normalized least squares:



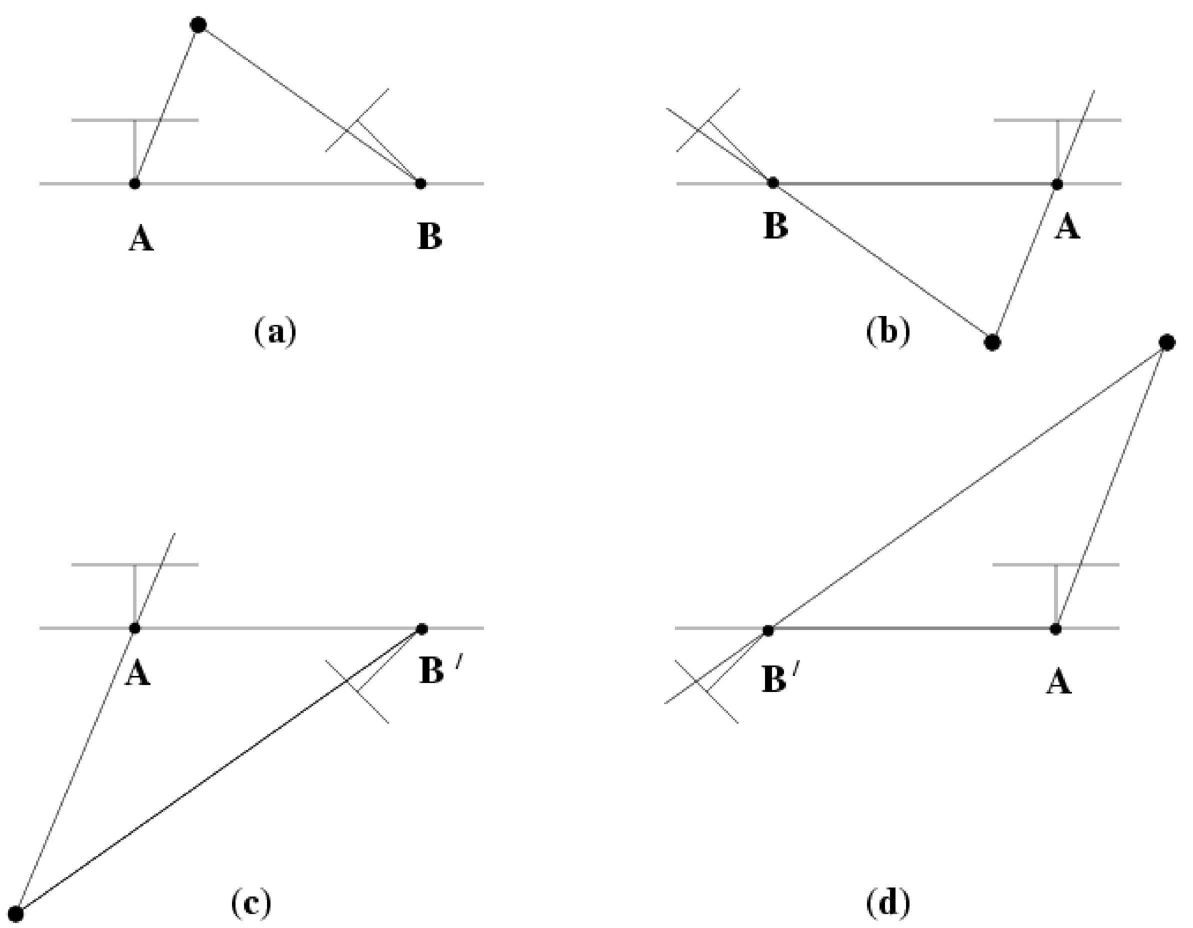
$$\begin{bmatrix} \frac{2}{700} & 0 & -1 \\ & \frac{2}{500} & -1 \\ & & 1 \end{bmatrix} \longrightarrow$$



Recovering the Pose

- Recover R, T from E
  - Only one solution where points is in front of both cameras
  - Apply motion consistency

$$E = T_{\times} R$$

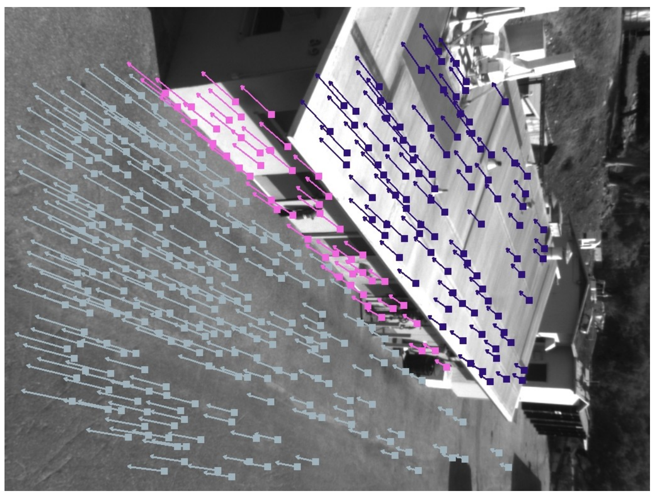


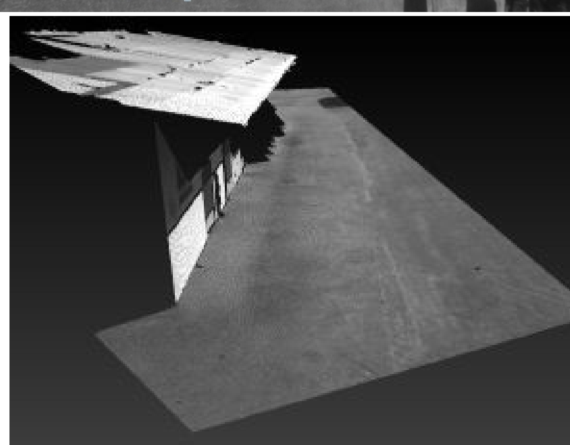
From 8 points to 5 points

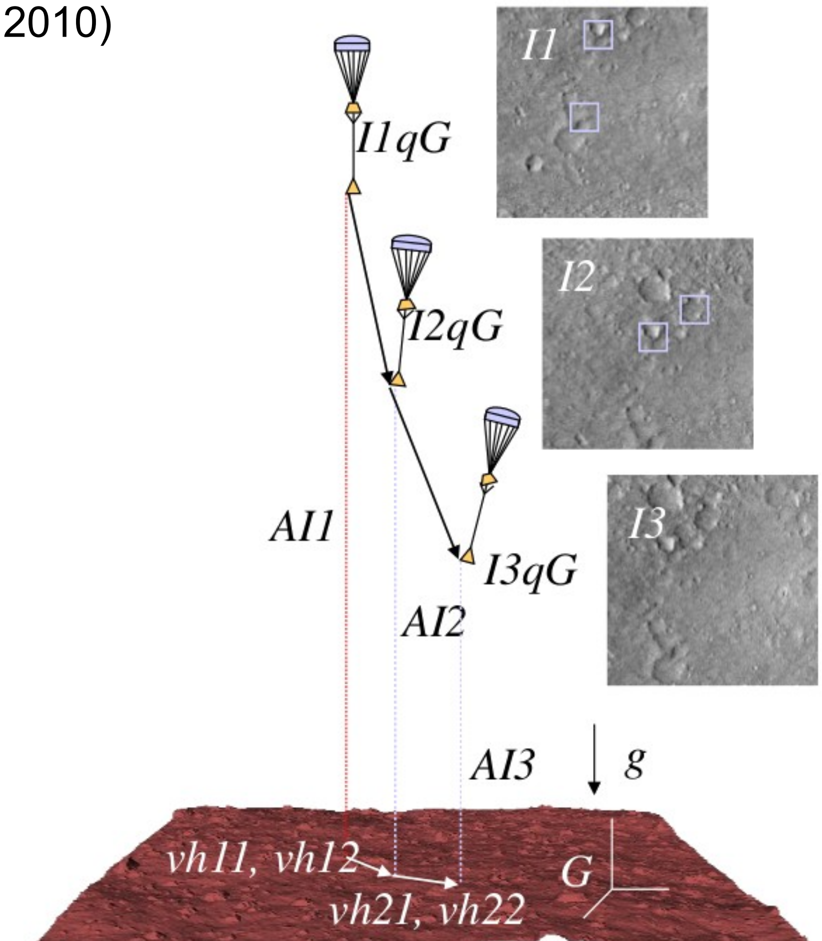
- Linear 8-point algorithm $Ae = 0$
  - Problem is only of dimension 5 (3 for rotation, 2 for translation up to scale)
  - Linear formulation is fast and simple to solve
- Non-linear 5-point algorithm (Nistér PAMI 204)
  - Finding roots of cubic polynomials
  - Mathematically hairy but fast implementations exist

Motion estimation with less than 5 points

- General case is a 5-dimensional problem
- Constraining the general case reduces the dimensionality:
  - Homography: Planar constraint, 4 points
    - Multi plane homography VO: Y. Cheng (ICRA 2010)

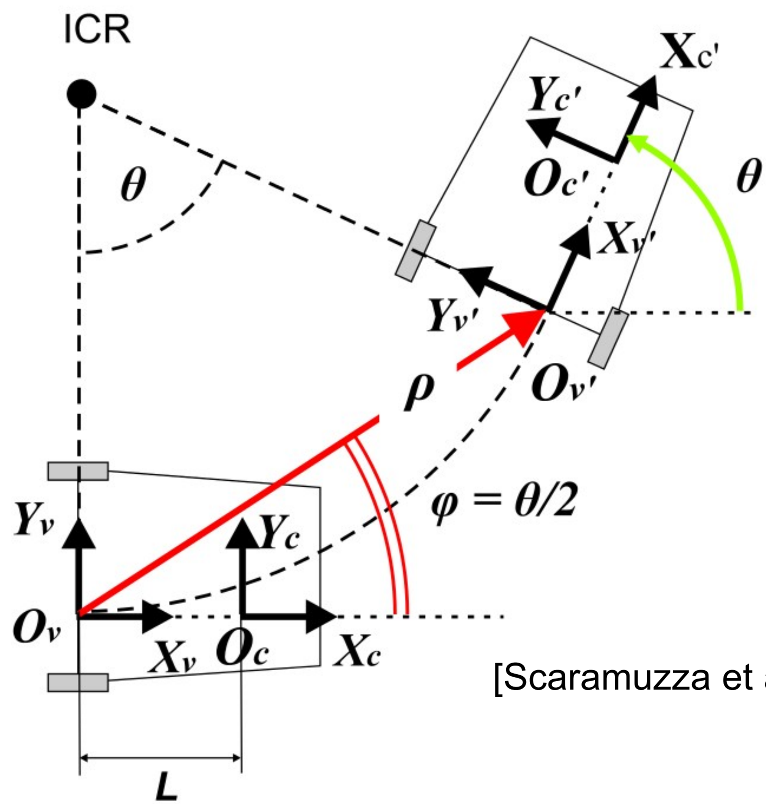






Motion estimation with less than 5 points

- General case is a 5-dimensional problem
- Constraining the general case reduces the dimensionality:
  - Using IMU for rotation: 2-dim constraint for translation up to scale [Weiss et al. ICRA 2012]
  - Using robot model and kinematics: 1 point [Scaramuzza et al. IJCV 2011]
  - Special case: known 3D coordinates of the points: stereo vision



[Scaramuzza et al. IJCV 2011]

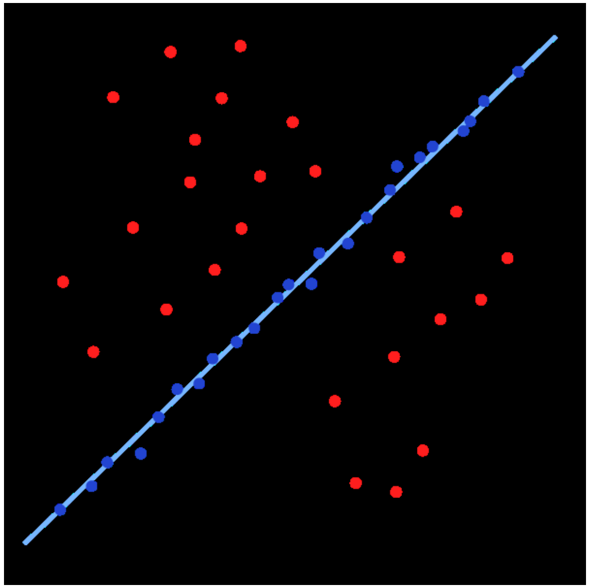
The RANSAC (RAndom SAmple Consensus) Algorithm for model fitting and outlier rejection

Assume:

- The model parameters can be estimated from N data items (e.g. essential matrix from 5-8 points)
- There are M data items in total.

The algorithm:

1. Select N data items at random
2. Estimate parameters (linear or nonlinear least square, or other)
3. Find how many data items (of M) fit the model with parameter vector within a user given tolerance, T. Call this k.
   if K is the largest (best fit) so far, accept it.
4. Repeat 1. to 4. S times



Questions:

- What is the tolerance?
- How many trials, S, ensure success?

The RANSAC (RAndom SAmple Consensus) Algorithm for model fitting

To ensure that RANSAC has high chance to find correct inliers, a sufficient number of trials must be executed. Let p be the probability of inliers of any given correspondence and P is a success probability after S trials. We have

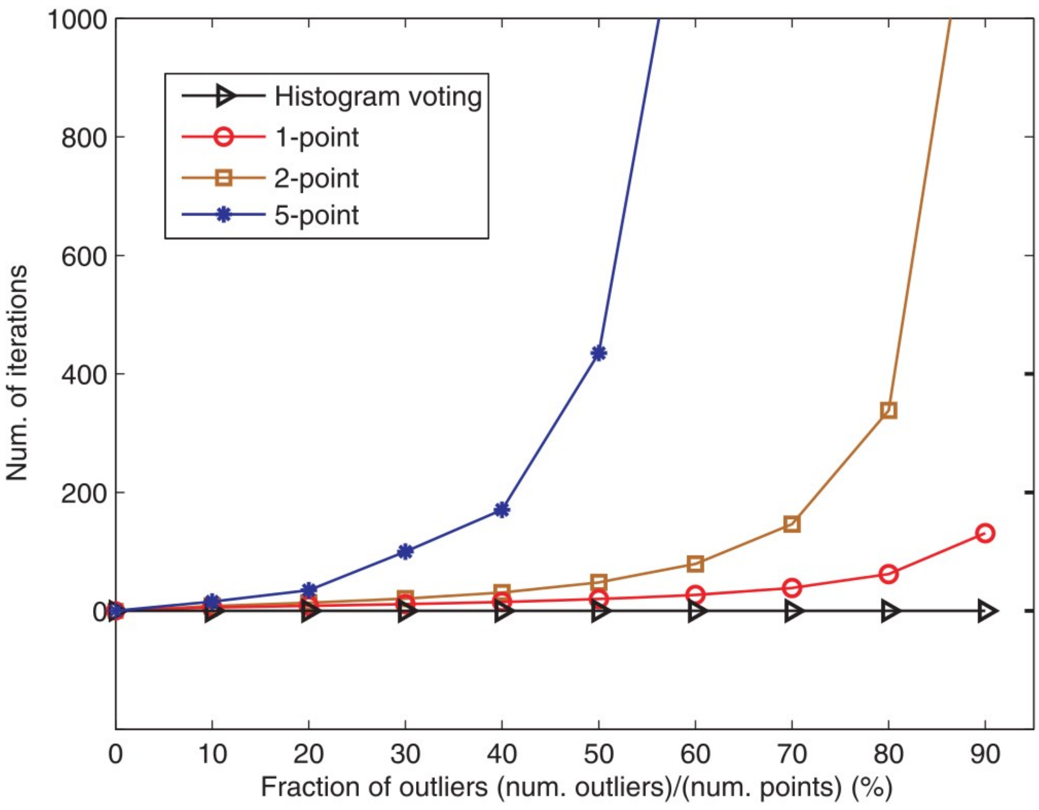
$$(1-P) = (1-p^k)^S$$

where p quickly decreases if many points are needed to fit the model!

And

$$S = \frac{\log(1-P)}{\log(1-p^k)}$$

Model fitting needs to be fast: this is executed at every camera frame!



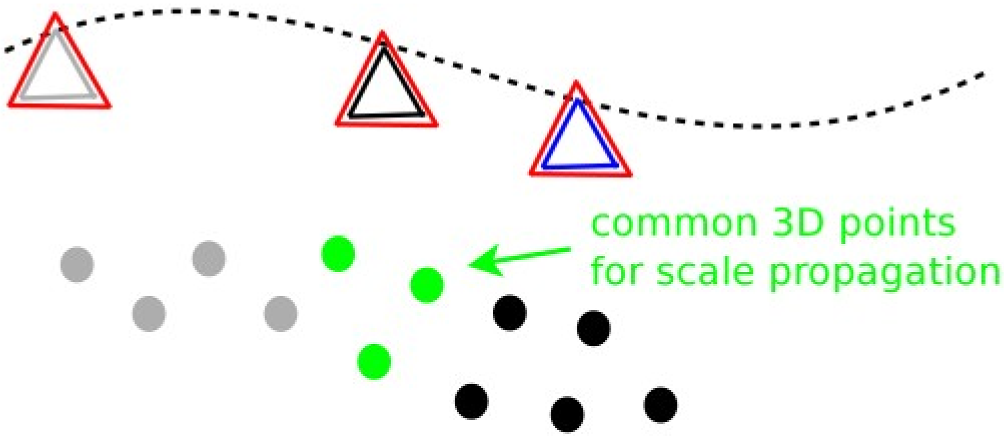
[Scaramuzza et al. IJCV 2011]

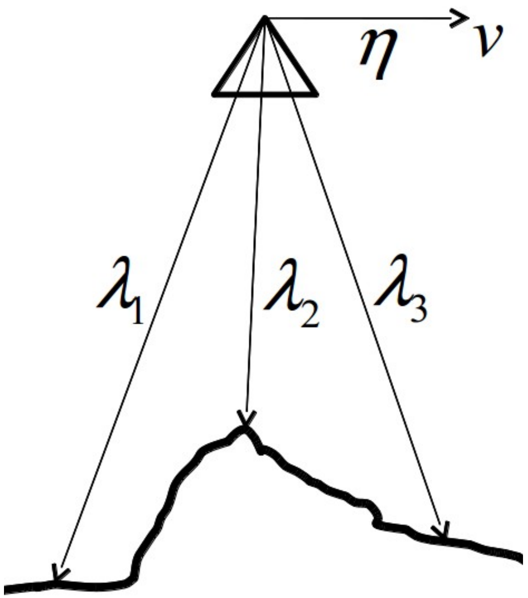
Scale propagation

- Scale of translation estimation between image pairs can vary arbitrarily

$$Ae = 0 = \lambda\, Ae = \Lambda\, Ae$$

  - Use common points to unify (propagate) the scale factor
  - Accumulated errors lead to scale drift





Getting to the point: image features

Jet Propulsion Laboratory
California Institute of Technology

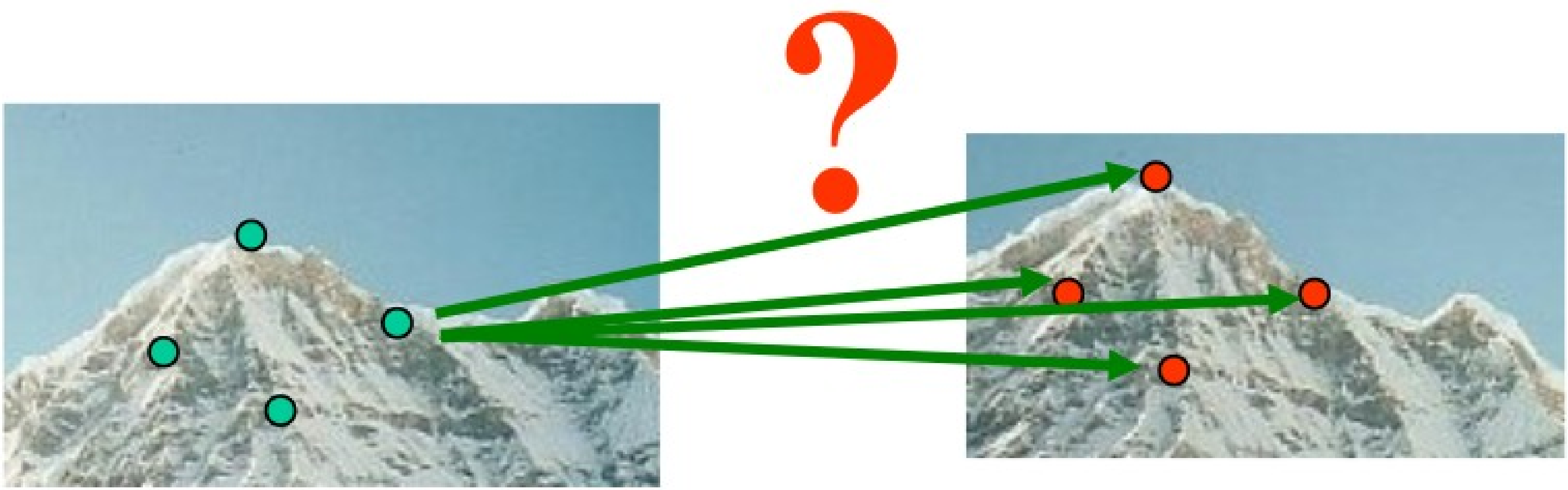
- Need at least 5 point correspondences in each image to determine general transformation
  - Extract salient points: feature detector
  - Detect the same salient points *independently* in both images

no chance to match!

We need a repeatable detector

Getting to the point: image features

- Need at least 5 point correspondences in each image to determine general transformation
  - Extract salient points: feature detector
  - Detect the same salient points *independently* in both images
  - Get sufficient information to recognize one point in the other image again



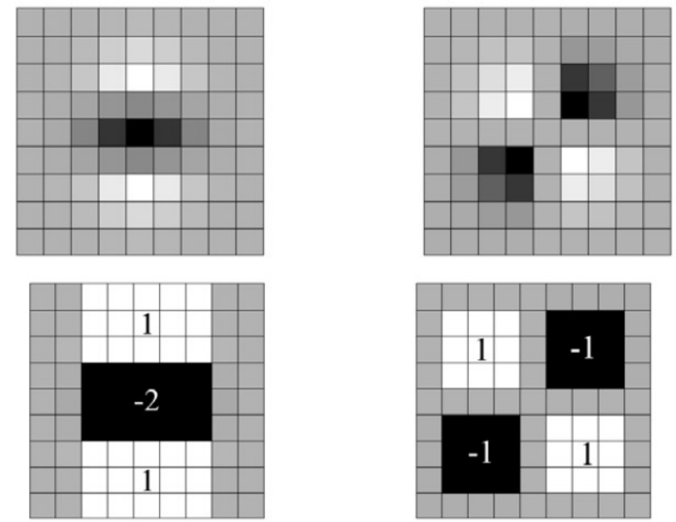
We need a reliable and distinctive descriptor

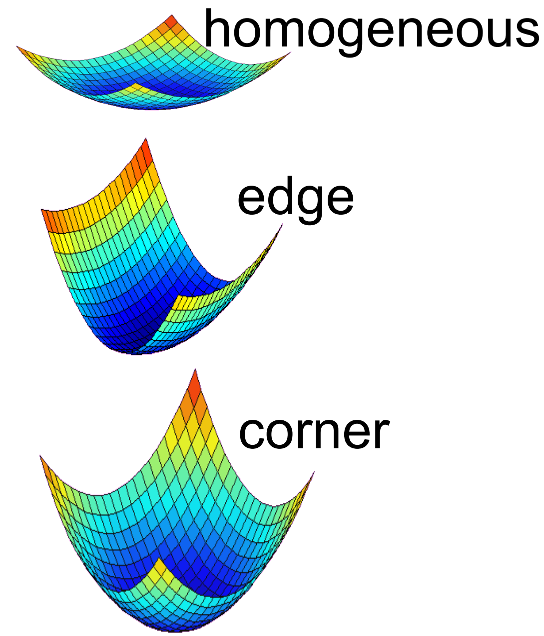
Getting to the point: Feature detectors

- Some examples:
  - FAST
  - AGAST
  - SIFT (DoG)
  - SURF (discretized DoG)

- General Idea:
  - Extract high contrast areas in the image
    - This often is at object borders: Parallax issue
  - Avoid edges

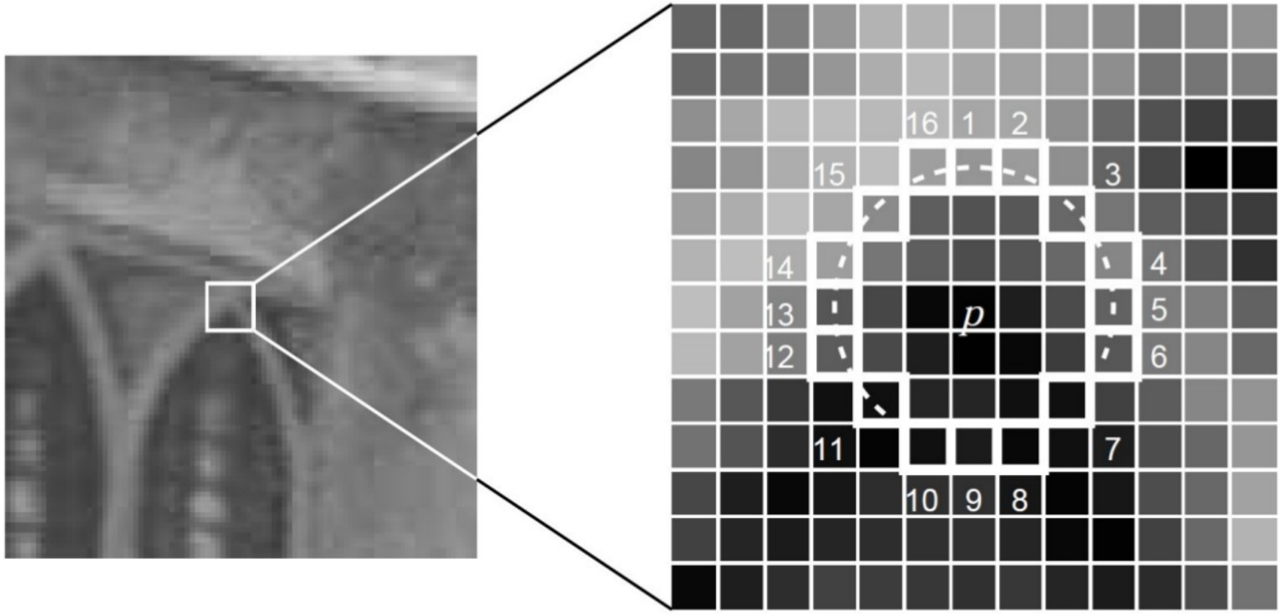
- Computaional complexity
  - Be as fast as possible:
    For every image 100s of features
  - Trade-off between high quality features
    (good repeatability) and computational complexity





Getting to the point: be FAST

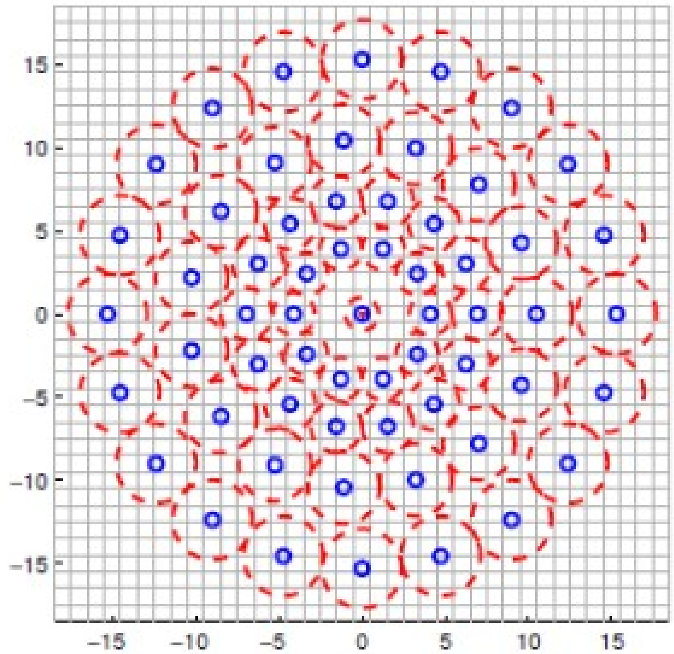
- Mostly used in real-time robotics applications
  - FAST/AGAST: on average checks 2.5 pixels per feature!



- Machine Learning was applied to
  - Generate a decision tree, that quickly discards pixels that are not a corner
  - Decision tree is build based on evaluating all 16 pixels and a training set
  - From the decision tree, C, Python or
  - Matlab code is generated (~6000 lines of code)
  - Available at: http://mi.eng.cam.ac.uk/~er258/work/fast.html

Keeping the Points: Tracking

- Idea: describe the region around a feature to find it again in the other image
- Examples of feature descriptors:
  - Image patch
  - SIFT
  - SURF
  - BRISK
  - DAISY
  - ......
- Should find the same feature again even if image is rotated, affine transformed, and scaled
- Any descriptor aims at a loss-less compression of the surrounding image patch including some invariances
- Apply normalization to mitigate illumination changes (e.g. ZM-SAD, ZM-SSD)

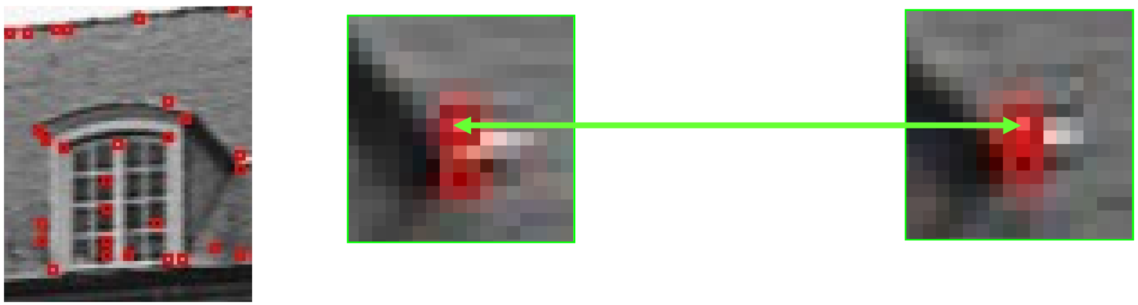


BRISK descriptor sampling pattern

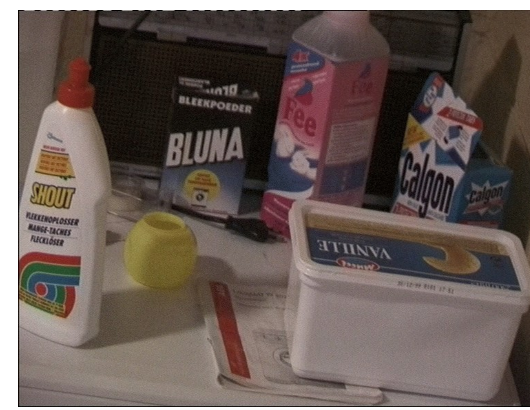
This is still in the kernel for motion estimation: happens 100s of times per frame

Keeping the point

- Issue with *-invariant descriptors:
  - Reduction in information

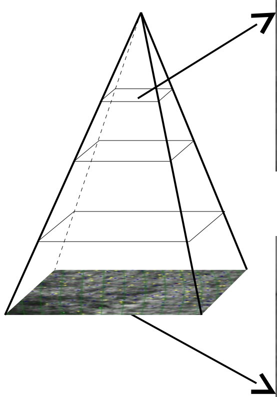


- Binary descriptors:
  - Further reduction of information
  - Very fast to compute
  - Issues with large datasets: Classification space is limited
  - Only abrupt class changes possible
  - Difficult to use for loop closures on large data sets



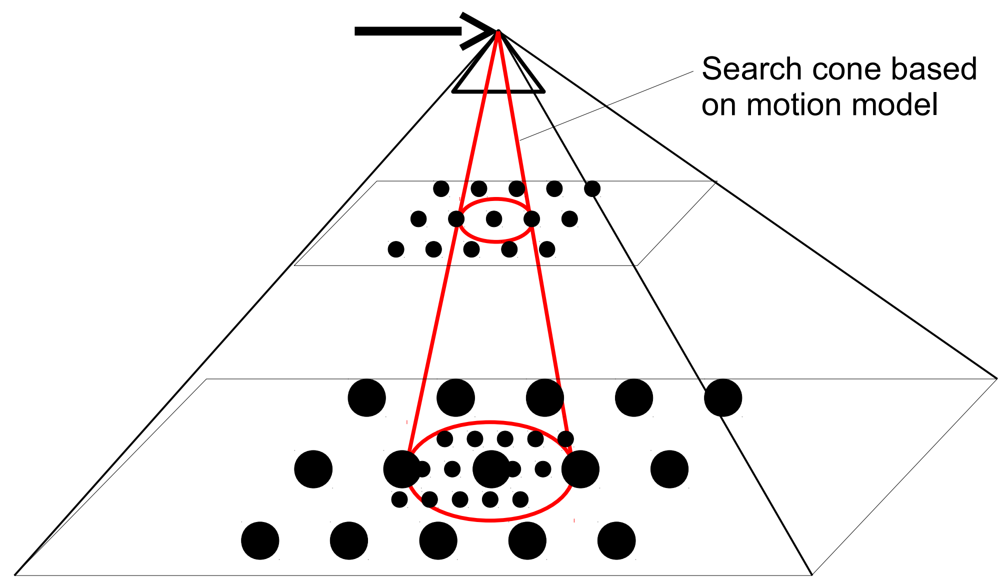
Keeping the Points: Speeding it up

- Feature tracking usually are the bottleneck in VO pipelines
  - Need to be done 100s of times per frame
- Constrain search region
  - Apply motion model (may include other sensors: IMU, GPS)
  - Use pyramidal approach:
    rough matching in low res image, refine in high res image
  - Combination of both (Klein & Murray, ISMAR 2007)



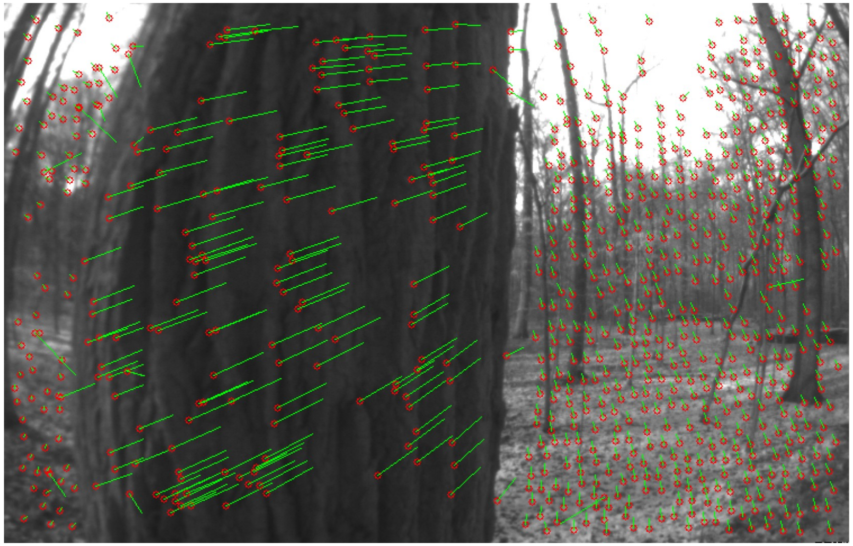
80x60 (Klein & Murray)

640x480 (Klein & Murray)



Keeping the Points: Speeding it up

- In VO: translation and rotation is usually small
  - This can be different for loop closing
- Sophisticated descriptors might be an overkill
- Plain image patches can be used as descriptors
  - Retains most information
  - 3x3 patches (8 pixels) can be computed efficiently by vector operations
  - Not even patch warping may be need
  - Search region must be kept very small!
- Robust outlier rejection often is preferred over robust and sophisticated feature matching



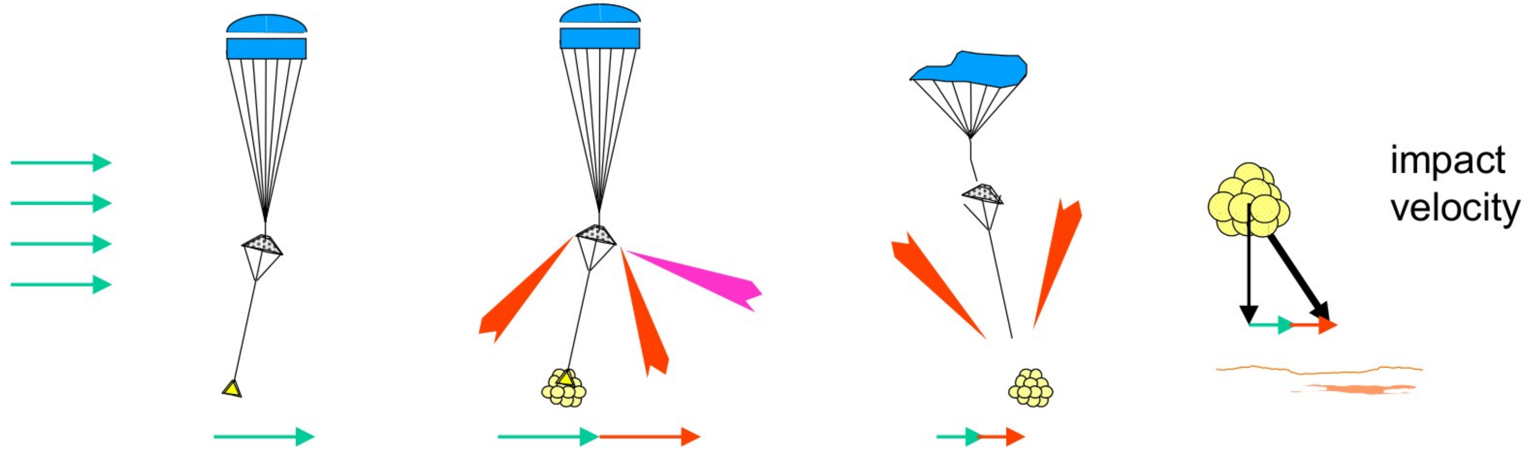
Optical flow computed with image patch matching and IMU motion model (Weiss et al. IROS13):

- 50Hz on 1 core of a cell phone 1.7GHz quadcore processor board

Putting All Together: Mars Exploration Rover (2003)

Jet Propulsion Laboratory
California Institute of Technology

- Not all robots drive/fly forever: some just need very good VO for some moments:
  Velocity estimation to land the Mars Exploration Rover

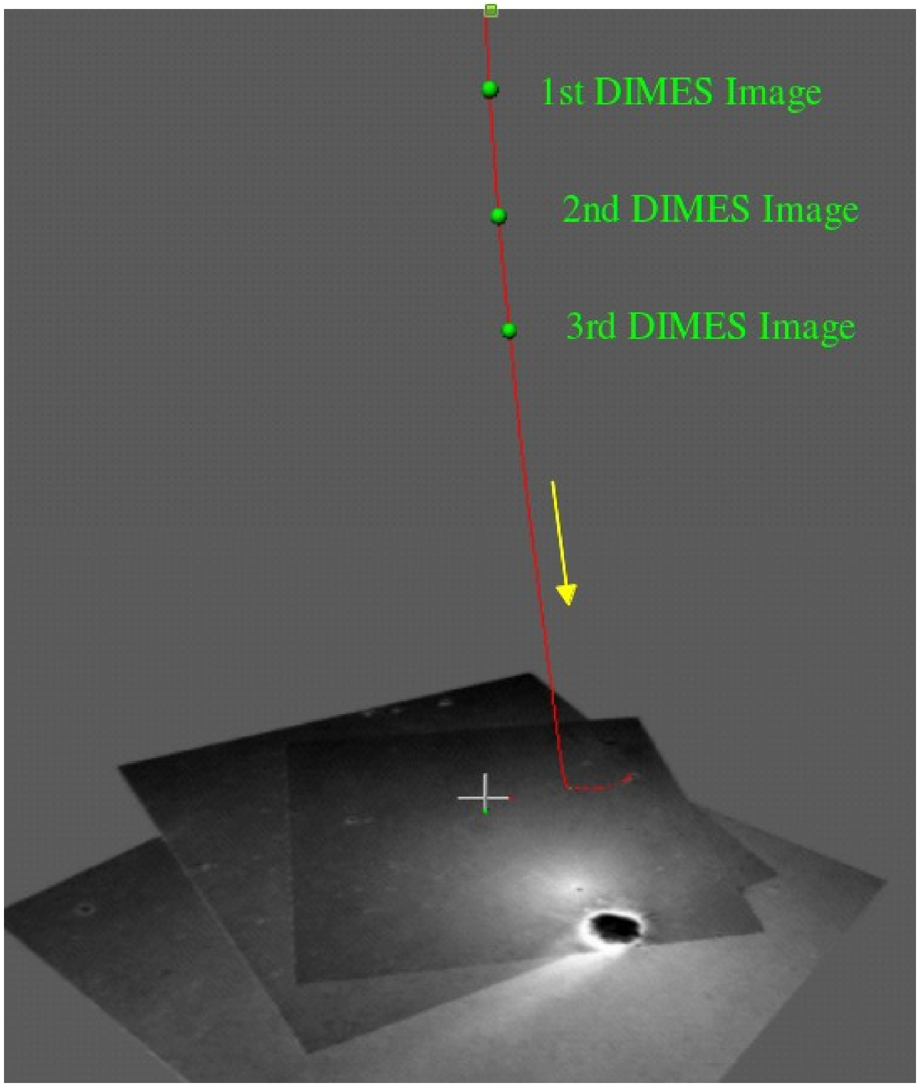


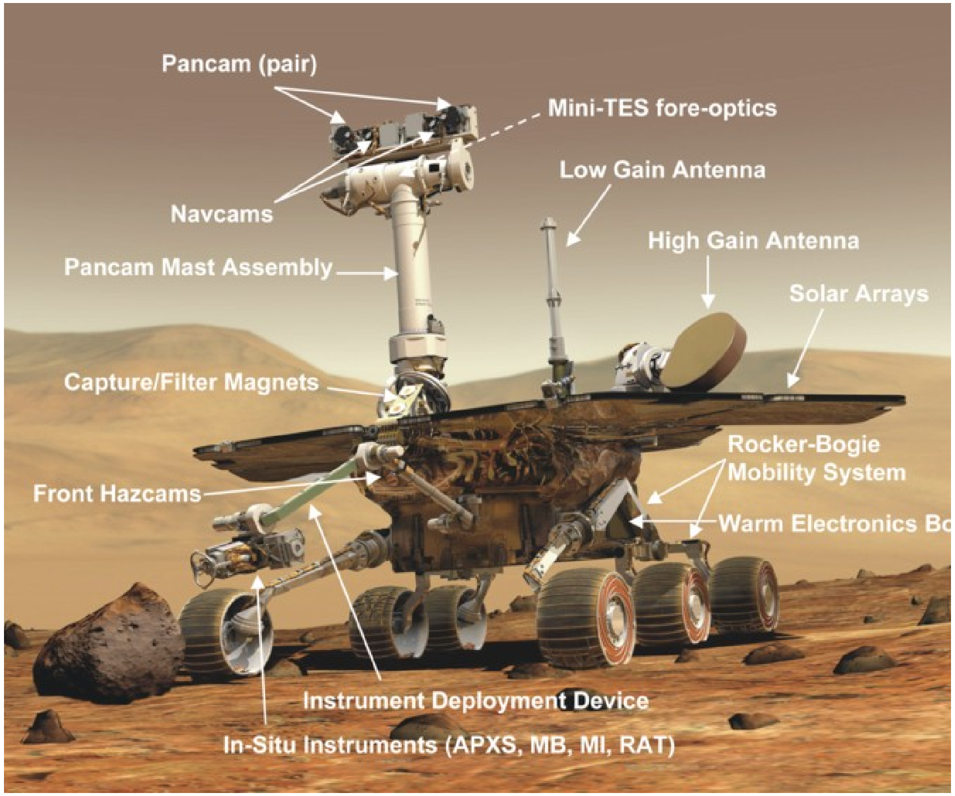
Efficient Implementation

Only template and window need to be rectified and flattened

- Computed on a coarse grid
- Homography assumption
- Application Region: Only computed in overlap region of images
- Sun direction parameter is used to mask out region around zero phase
- Parachute shadow

Putting All Together: Mars Exploration Rover (2003)





Q & A