Robust Sampling Consensus

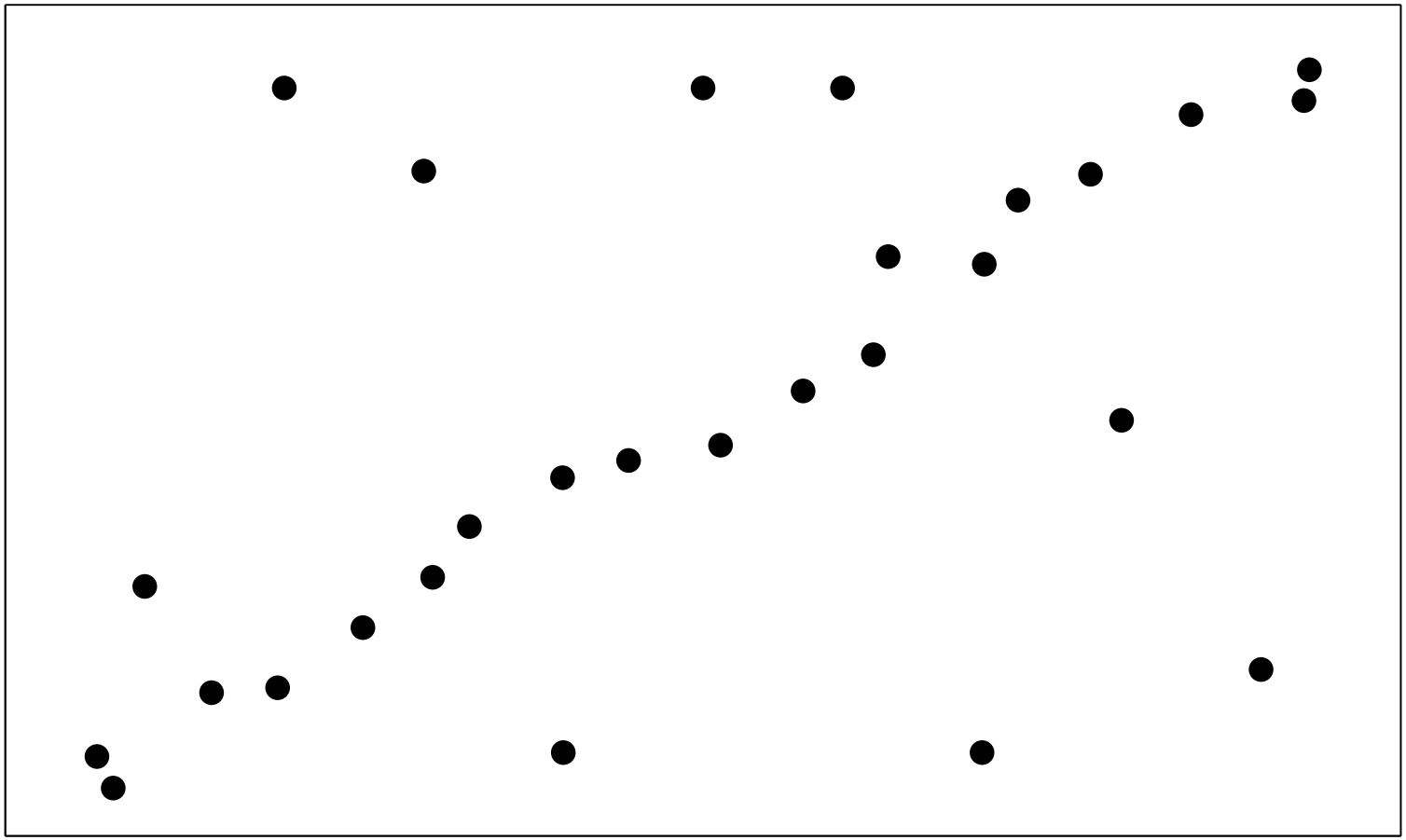
Karel Lebeda

January 24, 2013

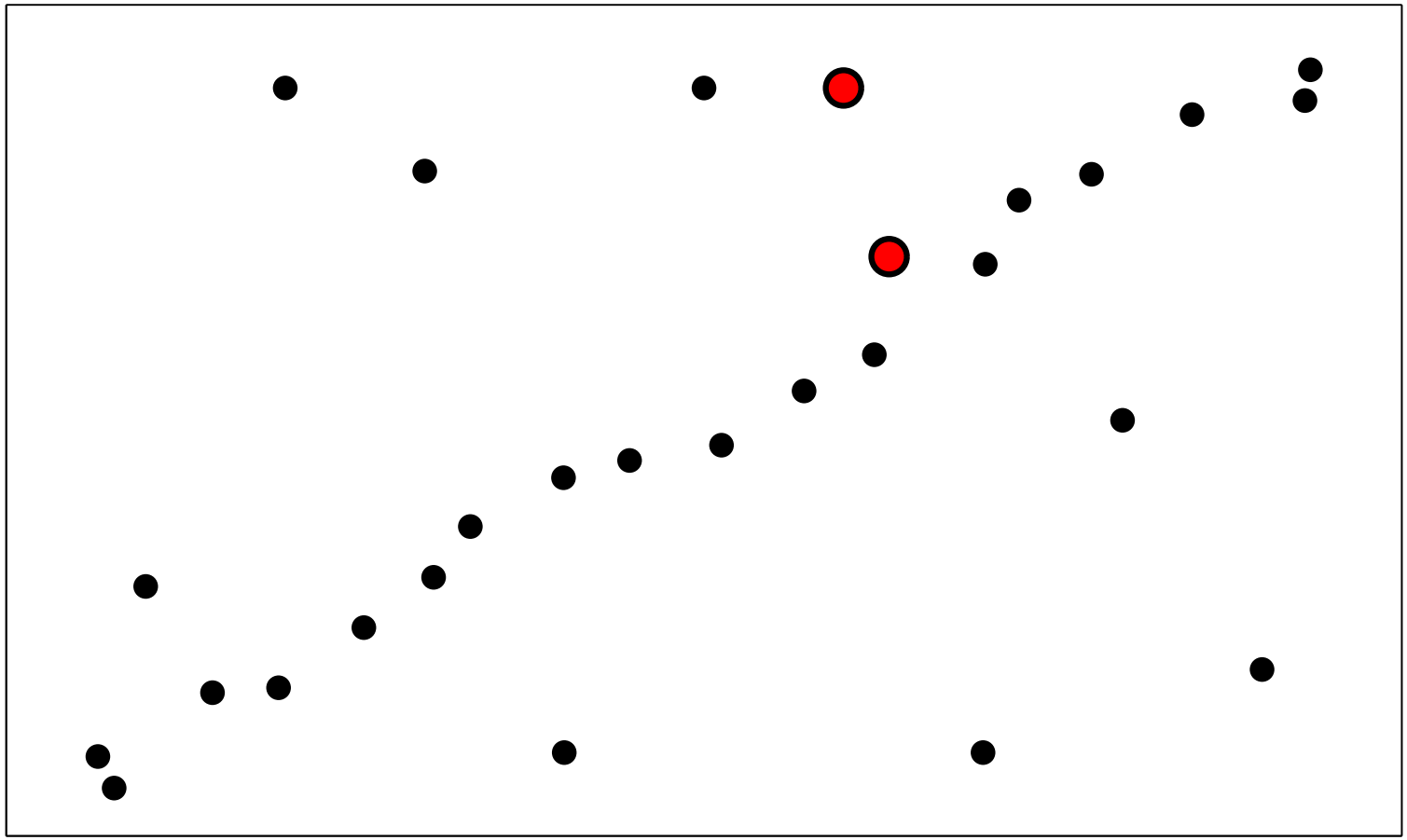
Motivation

- Geometric relations between cameras taking the same scene
- Useful for
  - 3D Reconstruction, SLAM, Image Stitching, Visual Tracking,...
- Input: noisy correspondences
- Need for a robust estimation
- Focus on random sampling methods

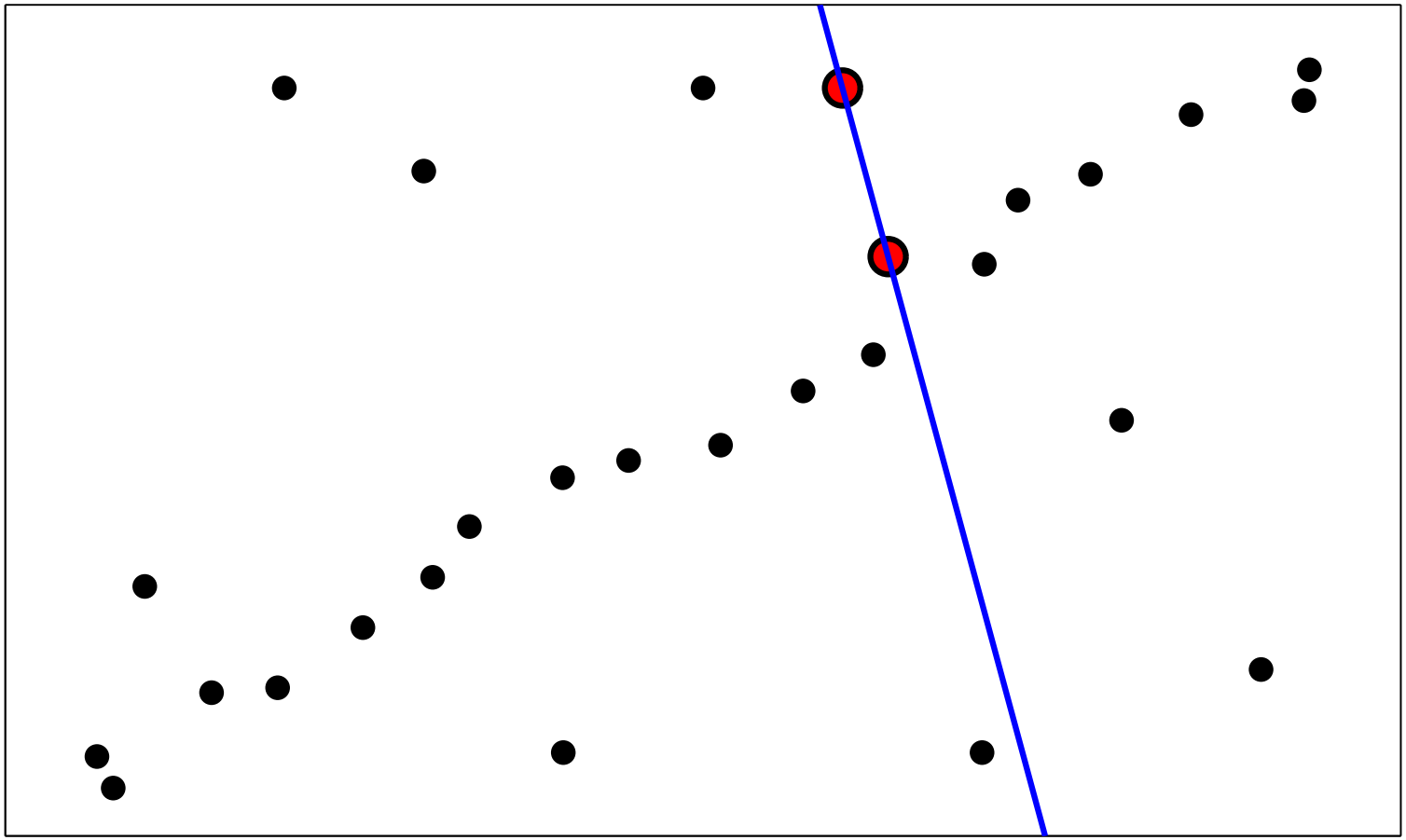
Random Sample Consensus



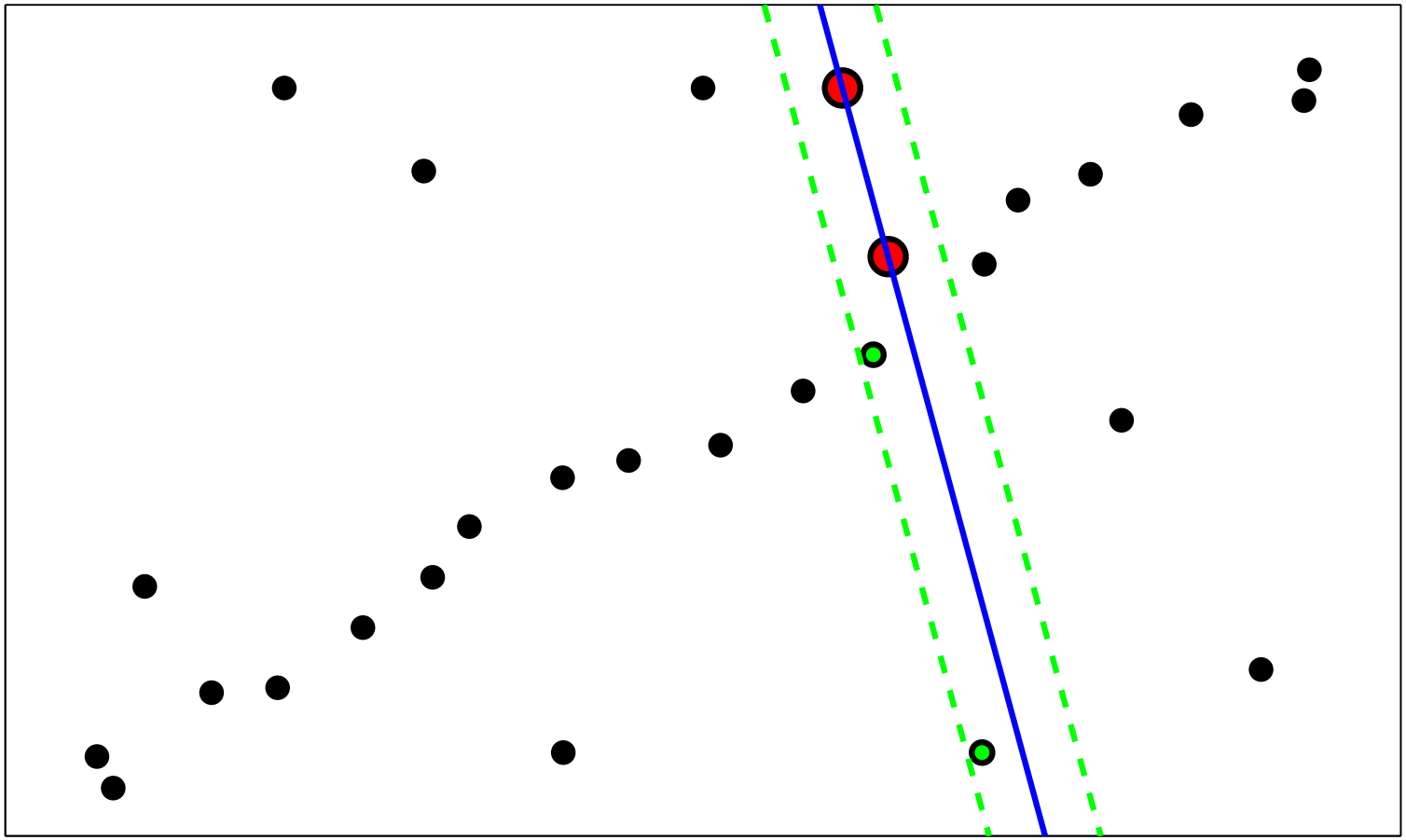
Random Sample Consensus



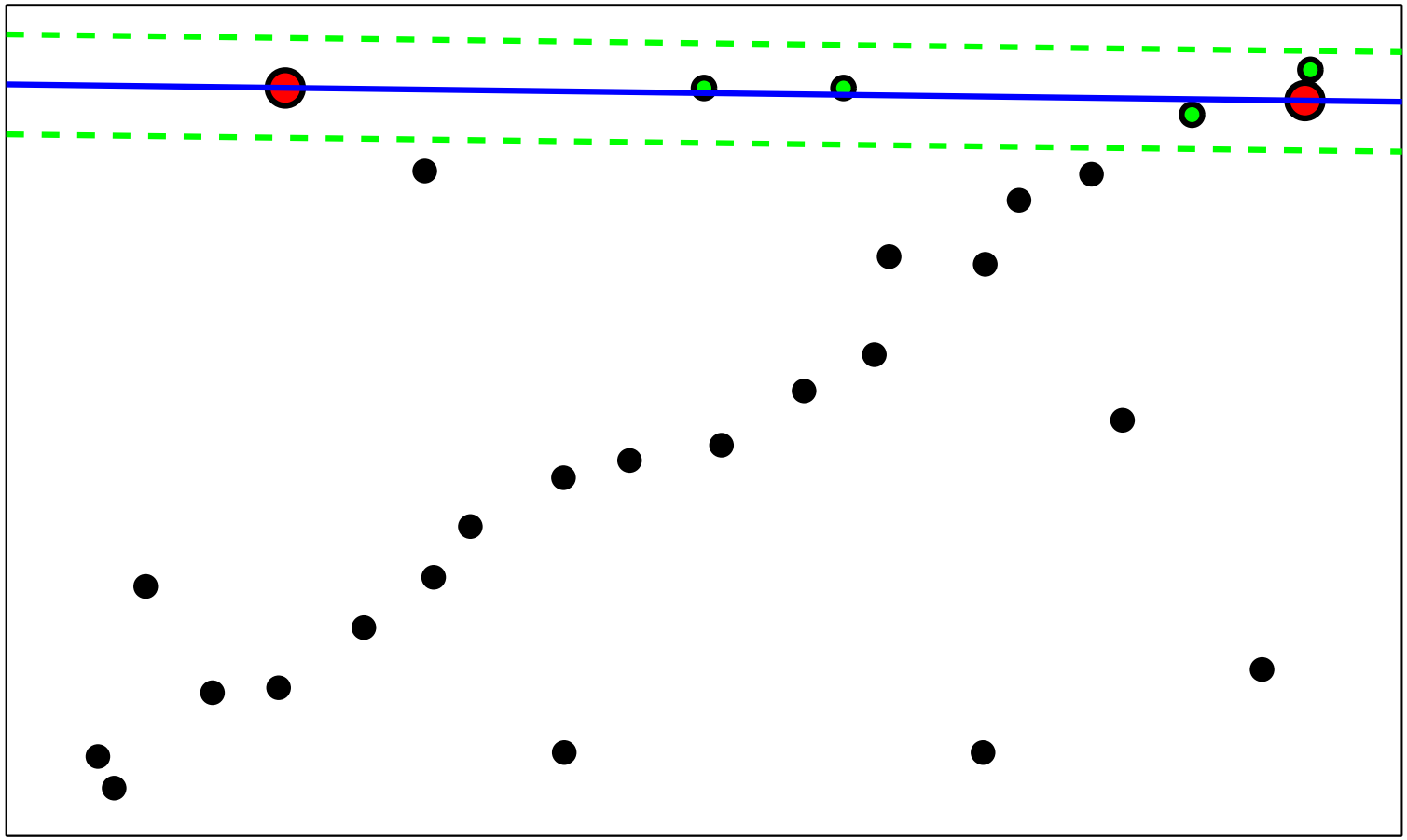
Random Sample Consensus



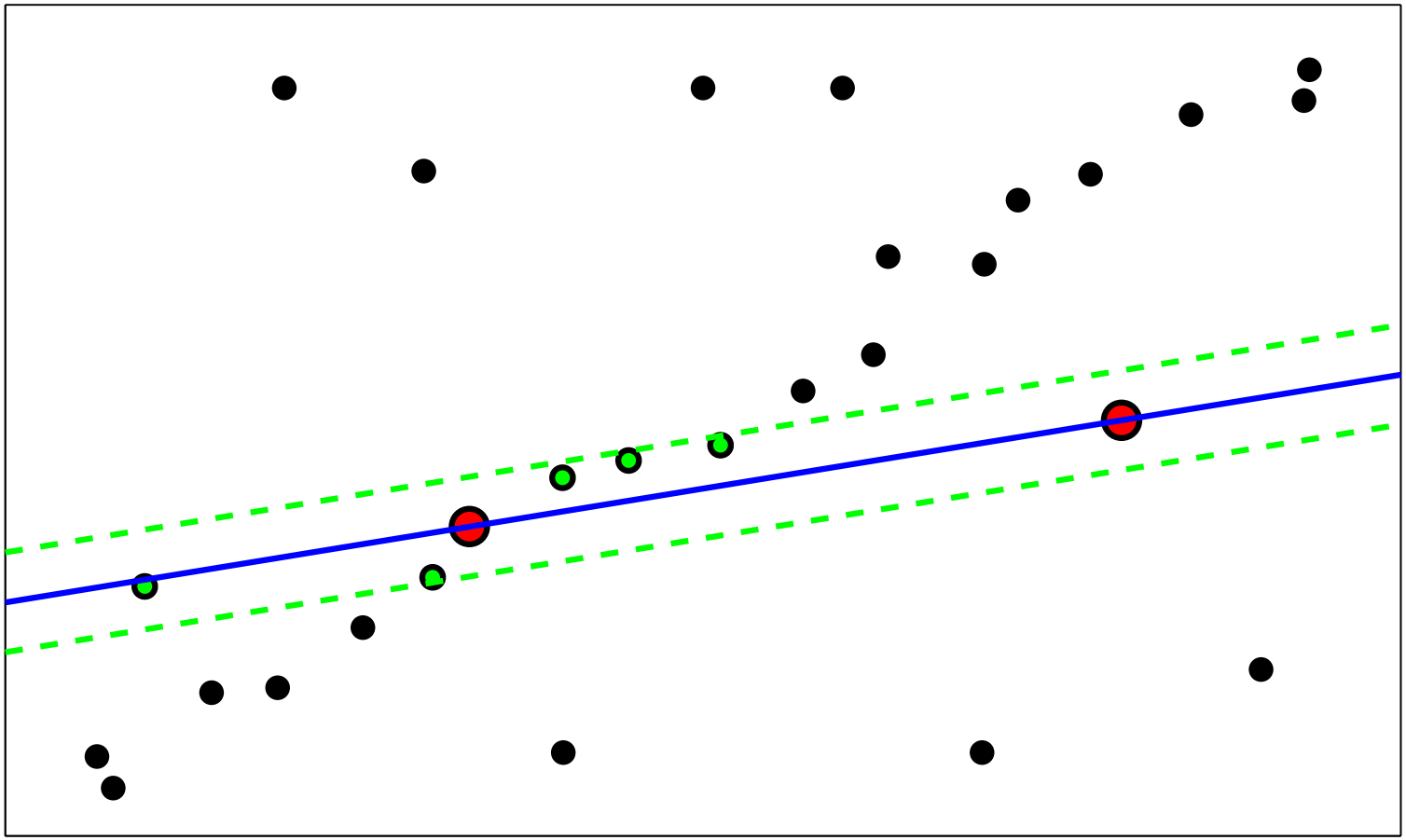
Random Sample Consensus



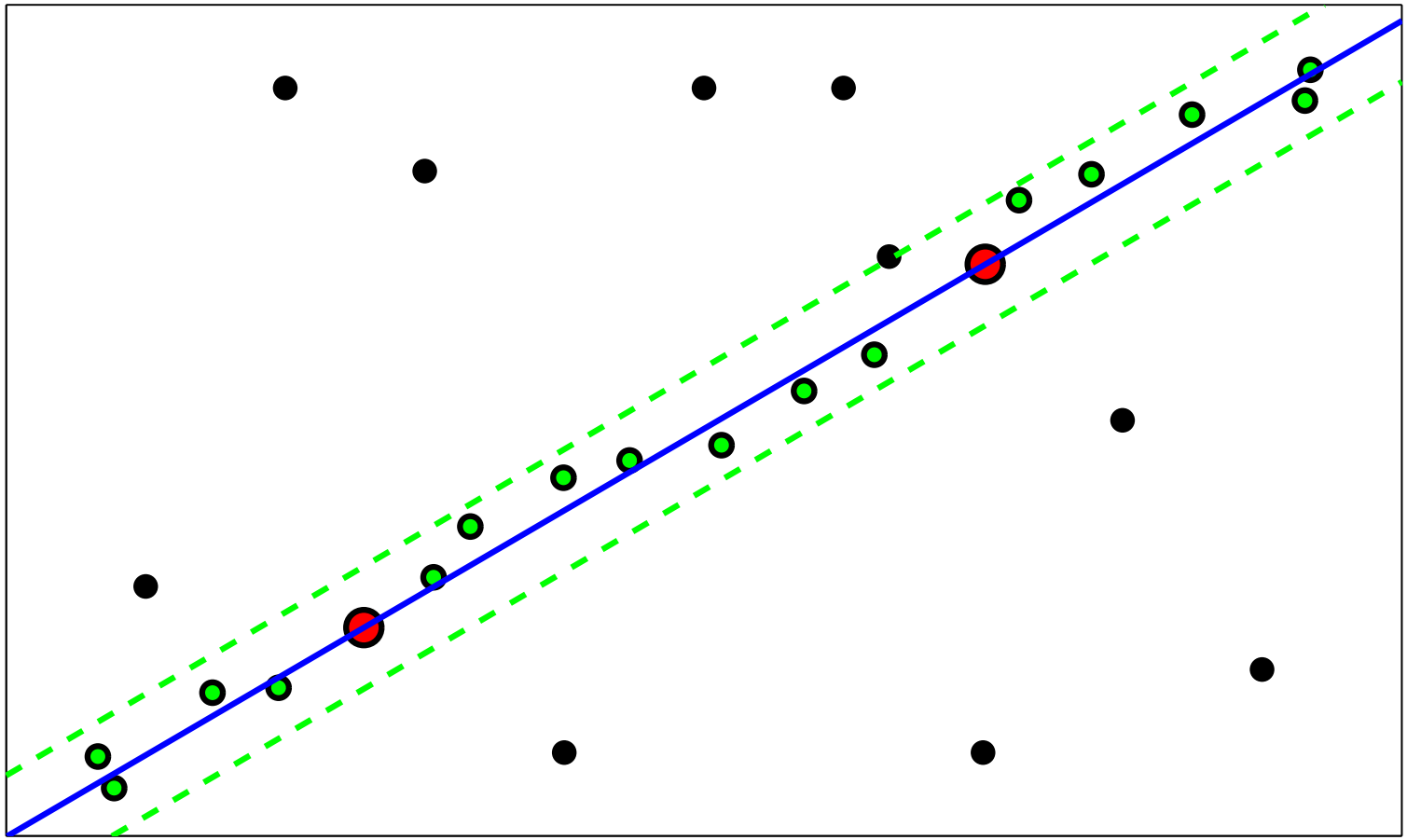
Random Sample Consensus



Random Sample Consensus

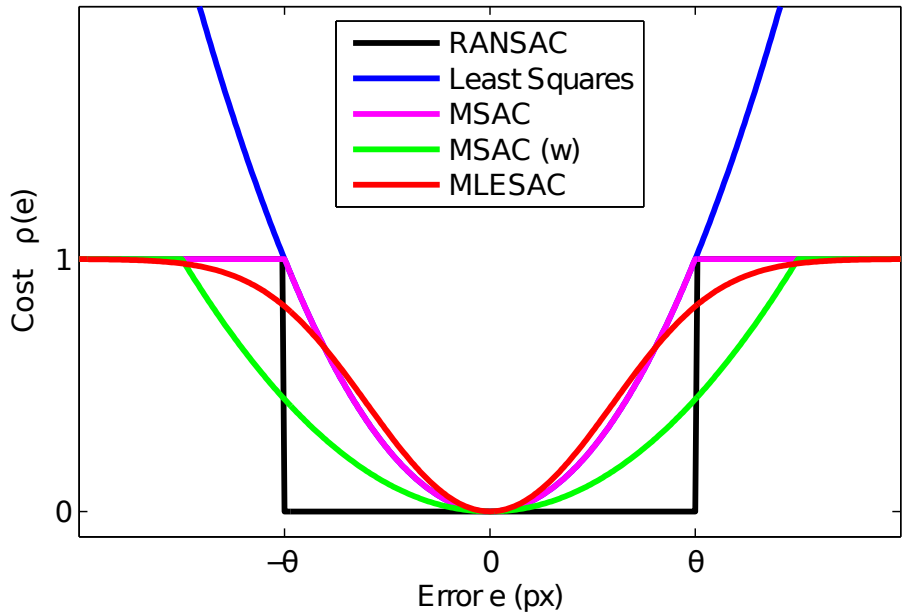


Random Sample Consensus



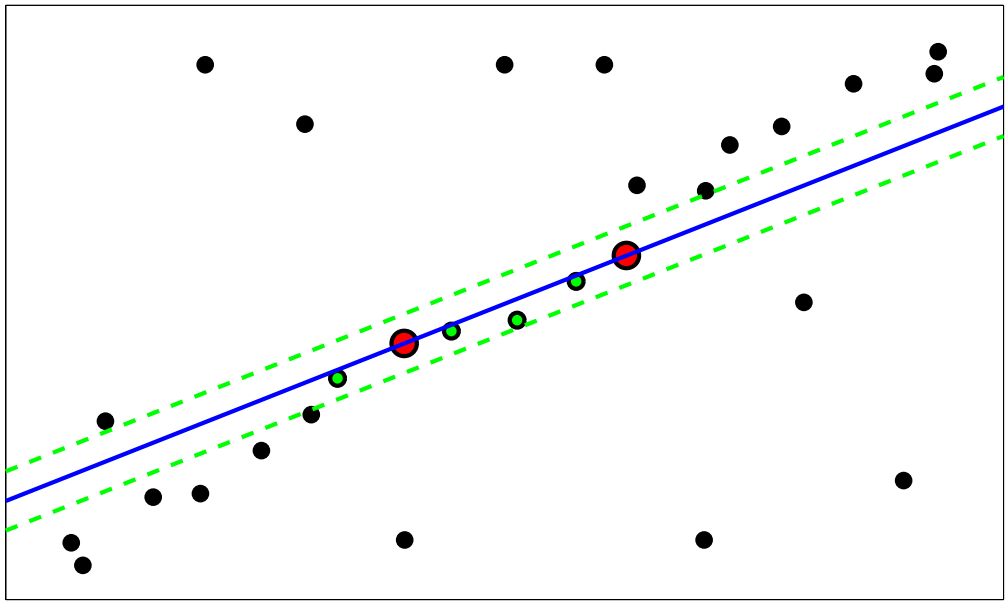
Cost Functions

- Selection of error scale
- Different cost functions – thresholding, truncating, log-likelihood,...



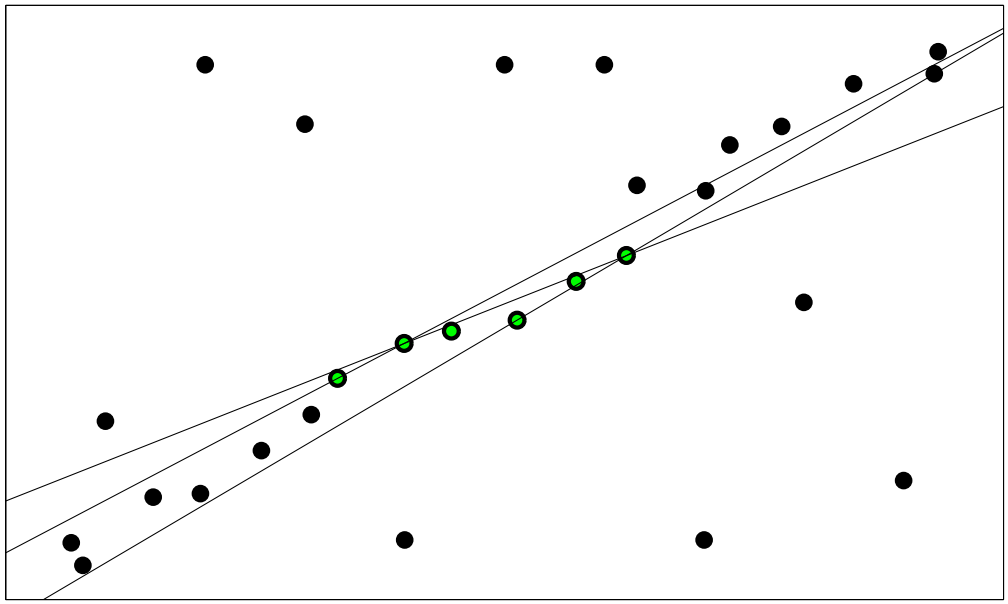
Local Optimization

- Refines promising samples
- Starts from all so-far-the-best samples
  $\Rightarrow$ probability $1/K$, totally $\log(K)$ LOs



Local Optimization

- Inner sampling, followed by iterative least squares
- Stabilizes results, decreases numbers of samples drawn



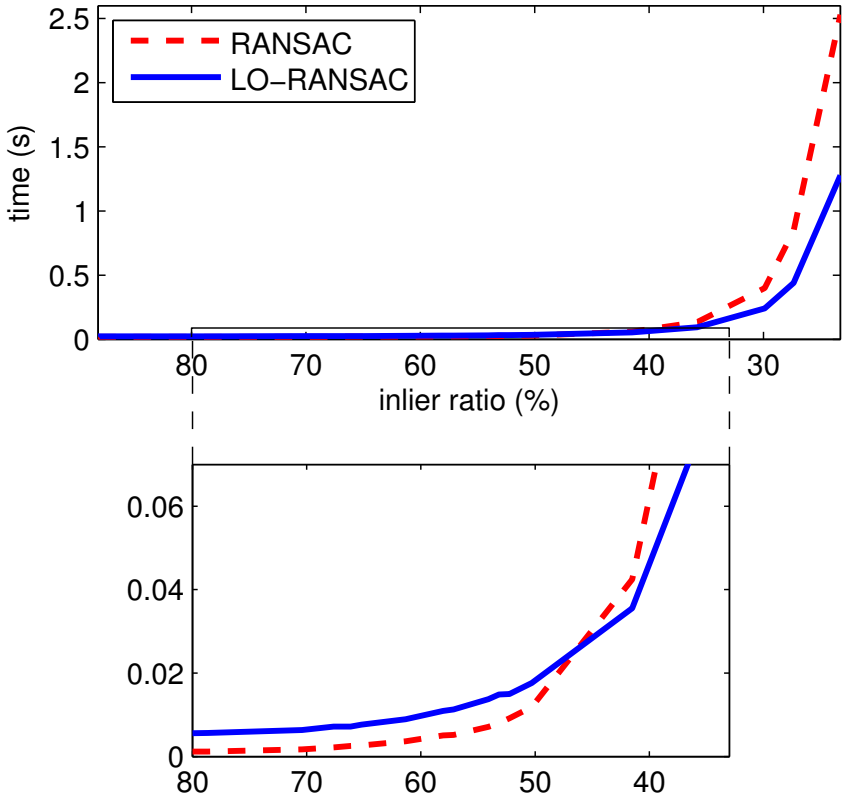
Major Contributions

- LO-RANSAC thoroughly analyzed and extensively tested
- Local Optimization speeded up
- Robustified to the error threshold selection
- Numerous implementation issues addressed

Speed of Local Optimization

$$t_{tot} = C_R \cdot K + C_{LO} \cdot \lceil \log(K) \rceil$$

- For small K,
  $C_{LO} \cdot \lceil \log(K) \rceil \gg C_R \cdot K$
- Even single run of the LO may take more time than the rest of RANSAC!

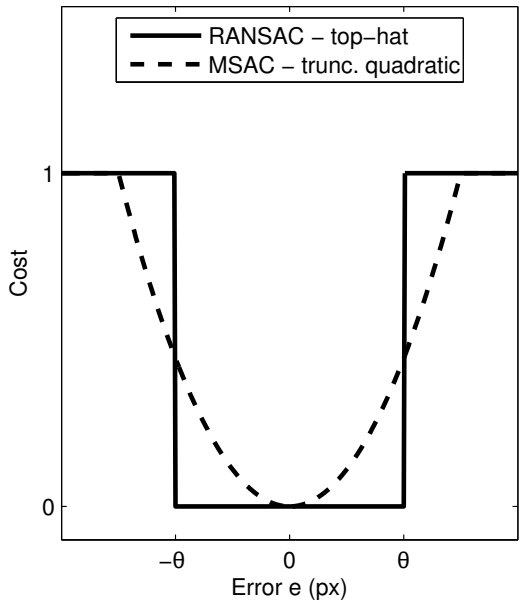


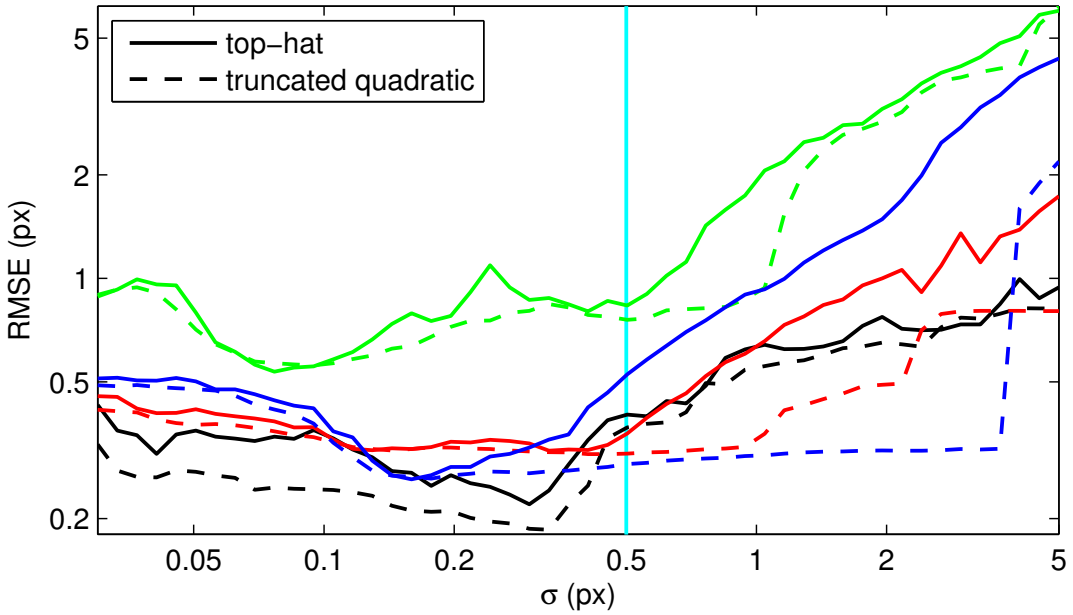
Execution Time Reduction

High execution time is caused by **repeated solutions** of sets of **linear equations**. Proposed speed-ups are:

- decrease the number of repetitions
  - simpler variant
  - hashing inlier sets
- decrease the number of equations
  - limiting number of inliers

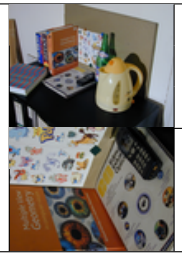
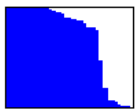
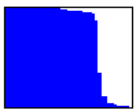
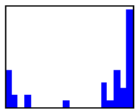
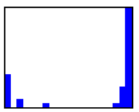
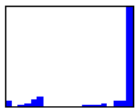
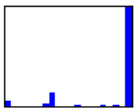
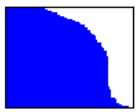
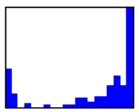
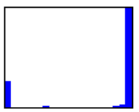
Cost Functions Analysis





- Similar performance at the optimal threshold, smooth ones much more robust to the threshold selection (colors code image pairs)

Example of Experimental Output

Solver→ Detectors→ Descriptors→			**M** MSER+ MSER- SIFT 10000 runs, $\sigma = 0.5$, conf = 95 %		**M.LO** MSER+ MSER- SIFT 10000 runs, $\sigma = 0.5$, conf = 95 %		
Image		Qty↓					
booksh		I	**28.8** ±1.7 (22-33)		**29.8** ±1.2 (23-33)		Inlss
		I (%)	**70.2** ±4.0 (54-80)		**72.7** ±2.8 (56-80)		
		Samp	**43.6** ±18.6 (11-139)		**40.5** ±14.4 (11-125)		
		$\text{Time}_{(ms)}$	**0.6** (NA)		**5.2** (NA)		H_{Inlss}
		Error	**3.13** ±4.41 (0.4-26.2)		**1.84** ±3.10 (0.4-25.8)		
		LO count	**0.0** ±0.0 (0-0)		**1.0** ±0.2 (1-4)		
box		I	**195.8** ±6.7 (177-222)		**204.3** ±12.1 (194-223)		Inlss
		I (%)	**84.8** ±2.9 (77-96)		**88.5** ±5.2 (84-97)		
		Samp	**8.8** ±1.7 (2-18)		**8.8** ±1.7 (2-18)		
		$\text{Time}_{(ms)}$	**0.5** (NA)		**8.5** (NA)		H_{Inlss}
		Error	**38.67**±26.49(0.7-111.6)		**43.70**±32.69(1.2-72.2)		
		LO count	**0.0** ±0.0 (0-0)		**1.0** ±0.0 (1-1)		
castle		I	**110.0** ±7.3 (90-124)		**122.8** ±2.0 (105-124)		Inlss
		I (%)	**71.4** ±4.8 (58-81)		**79.7** ±1.3 (68-81)		
		Samp	**41.2** ±17.6 (12-130)		**37.6** ±12.3 (12-130)		
		$\text{Time}_{(ms)}$	**1.0** (NA)		**7.5** (NA)		H_{Inlss}
		Error	**4.41** ±6.82 (0.3-59.9)		**0.84** ±1.68 (0.4-16.4)		
		LO count	**0.0** ±0.0 (0-0)		**1.0** ±0.0 (1-1)		

Outputs

- Main contributions published in paper:
  Lebeda, Matas, Chum: **Fixing the Locally Optimized RANSAC**.
  In *Proceedings of the British Machine Vision Conference*, 2012.
  + technical report with extended experimental evaluation
- Implementation available under GNU GPL:
  `http://cmp.felk.cvut.cz/software/LO-RANSAC/`
- Used datasets available (including newly created GT points):
  `http://cmp.felk.cvut.cz/data/geometry2view/`