Monocular SLAM and Real-Time Scene Perception

Andrew Davison
Robot Vision Group
Department of Computing
Imperial College London

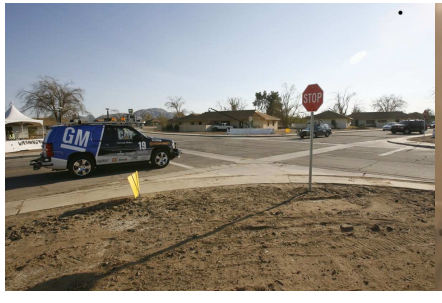
September 7, 2012

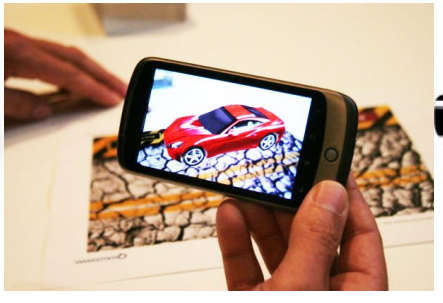
Robot Vision

- Robot Vision: in the real world; in a real-time loop; and for a purpose.

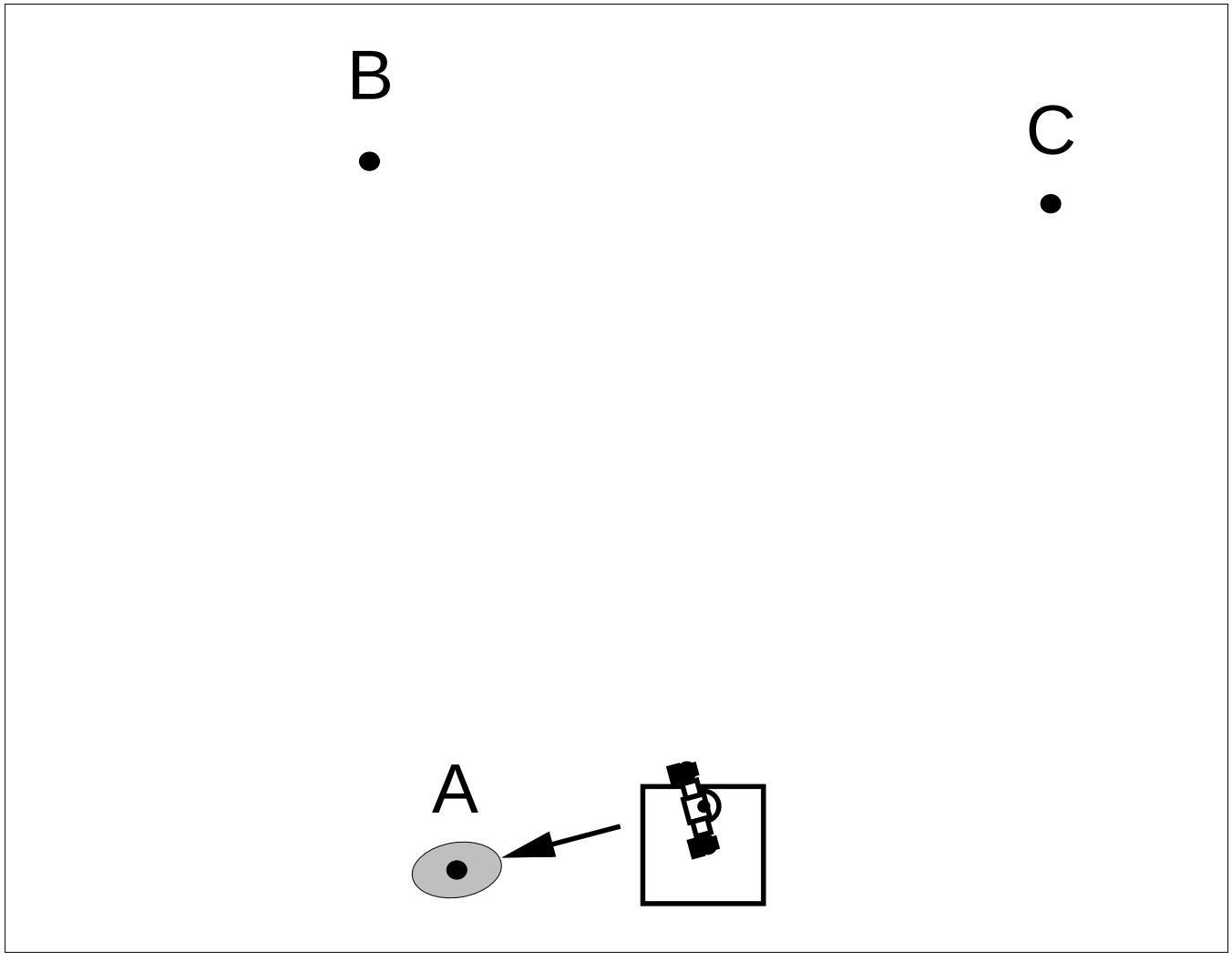
Performance in robot vision is advancing *fast*. What are the reasons?

- Continued exponential increase in low-cost computer power.
- Common understanding of key principles of inference under uncertainty.
- A wealth of tools that *really work* are publicly available as algorithms or code and can be easily put together into systems.



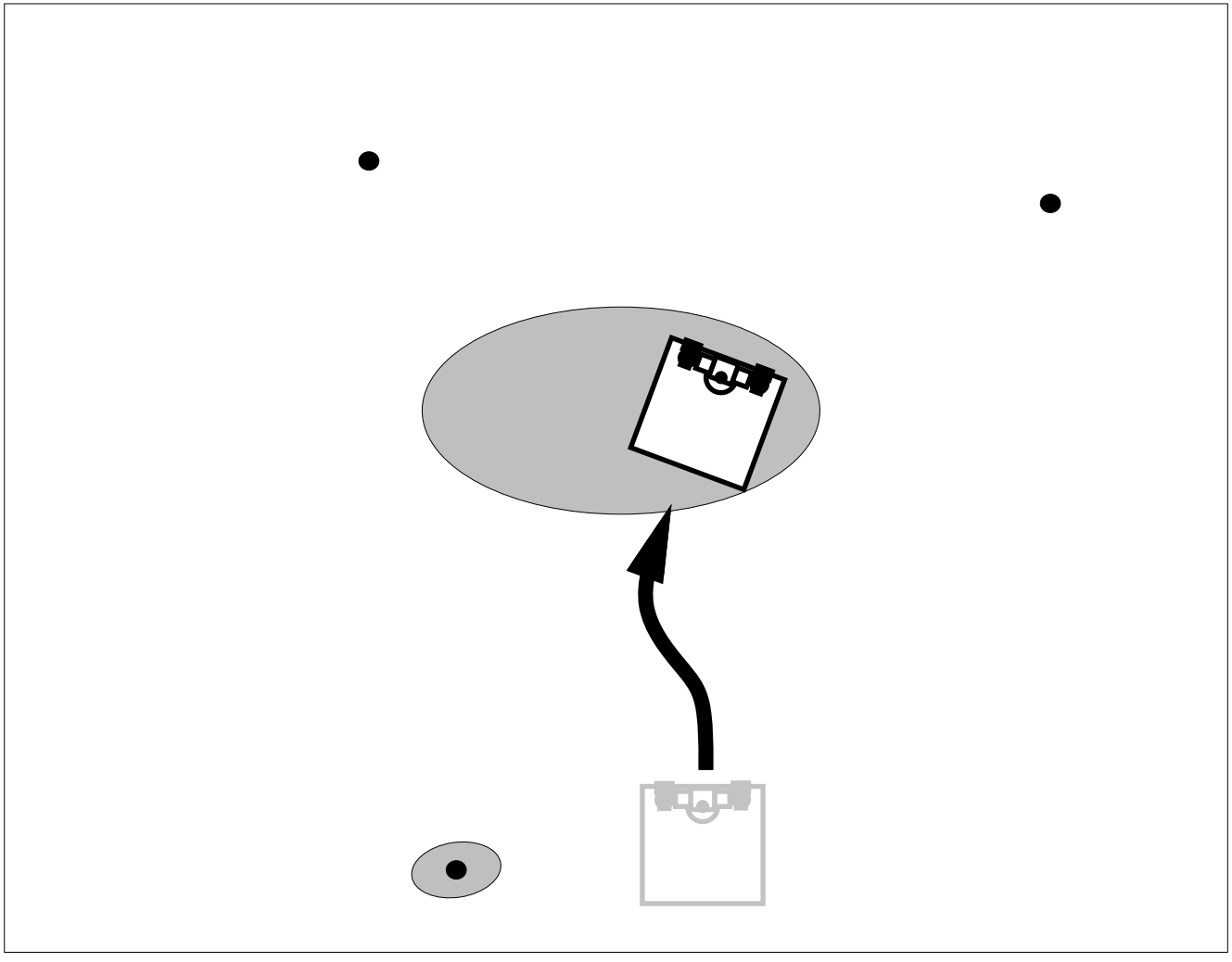




Simultaneous Localisation and Mapping



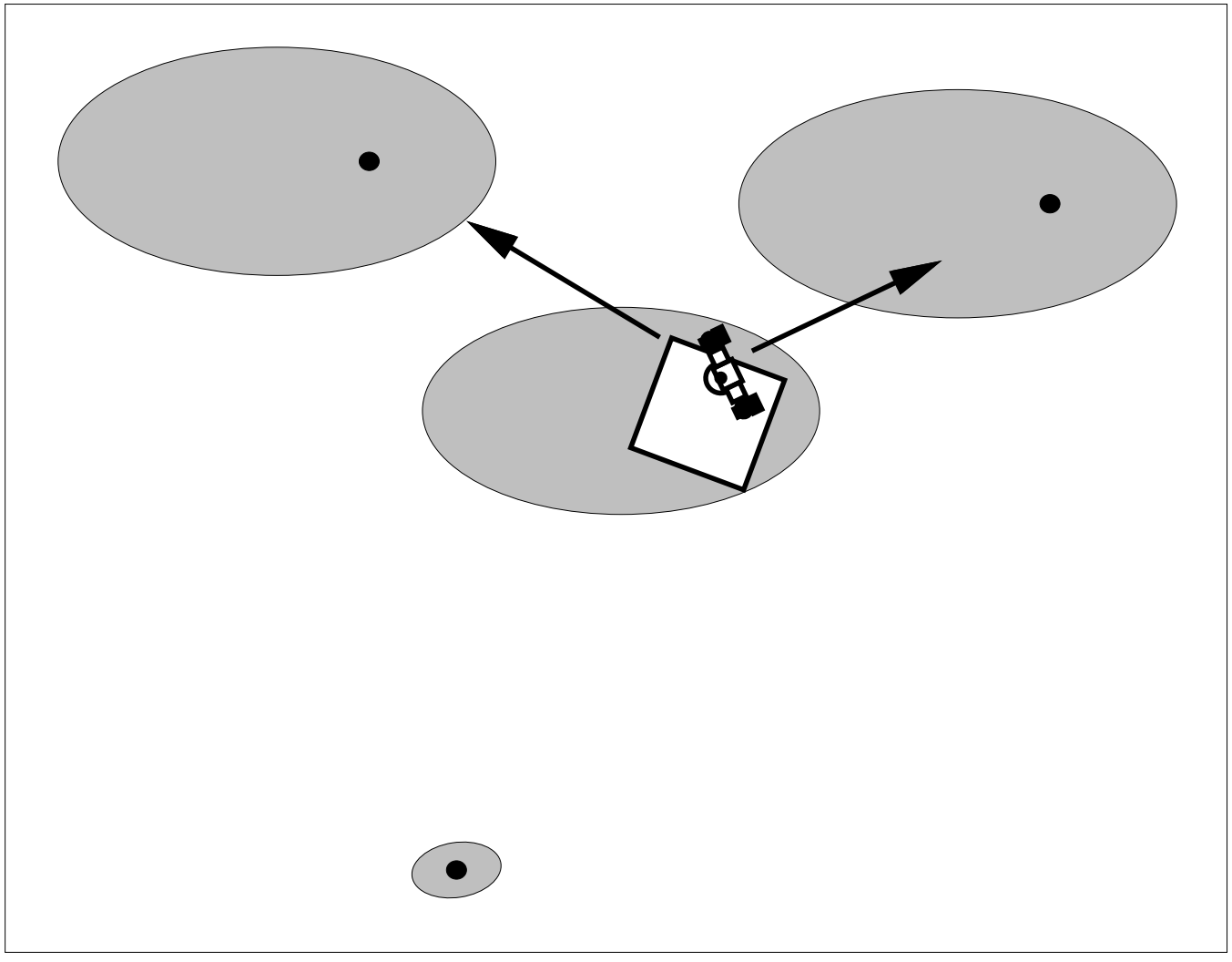
(a) Robot start (zero uncertainty); first measurement of feature A.

Simultaneous Localisation and Mapping



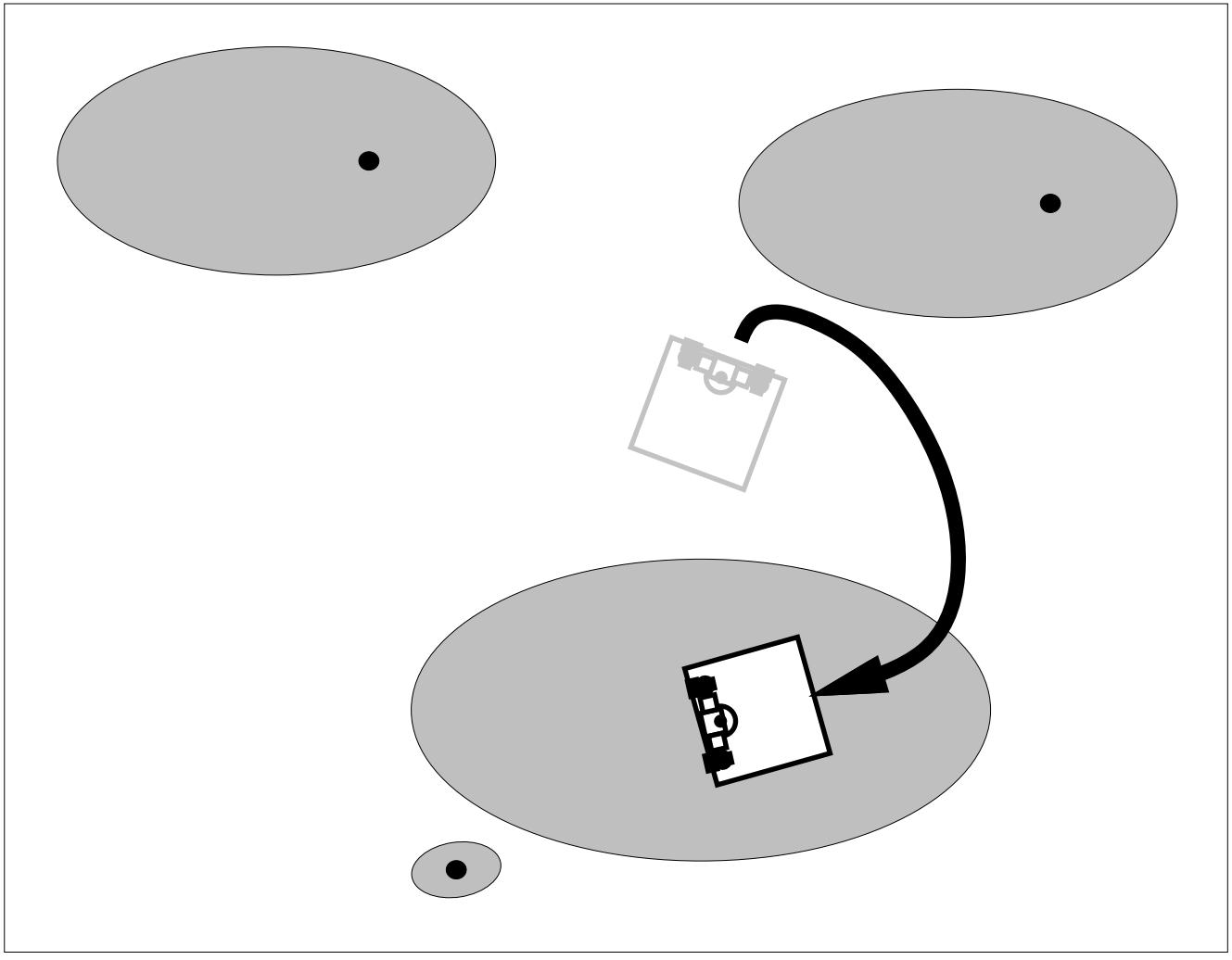
(b) Robot drives forwards (uncertainty grows).

Simultaneous Localisation and Mapping



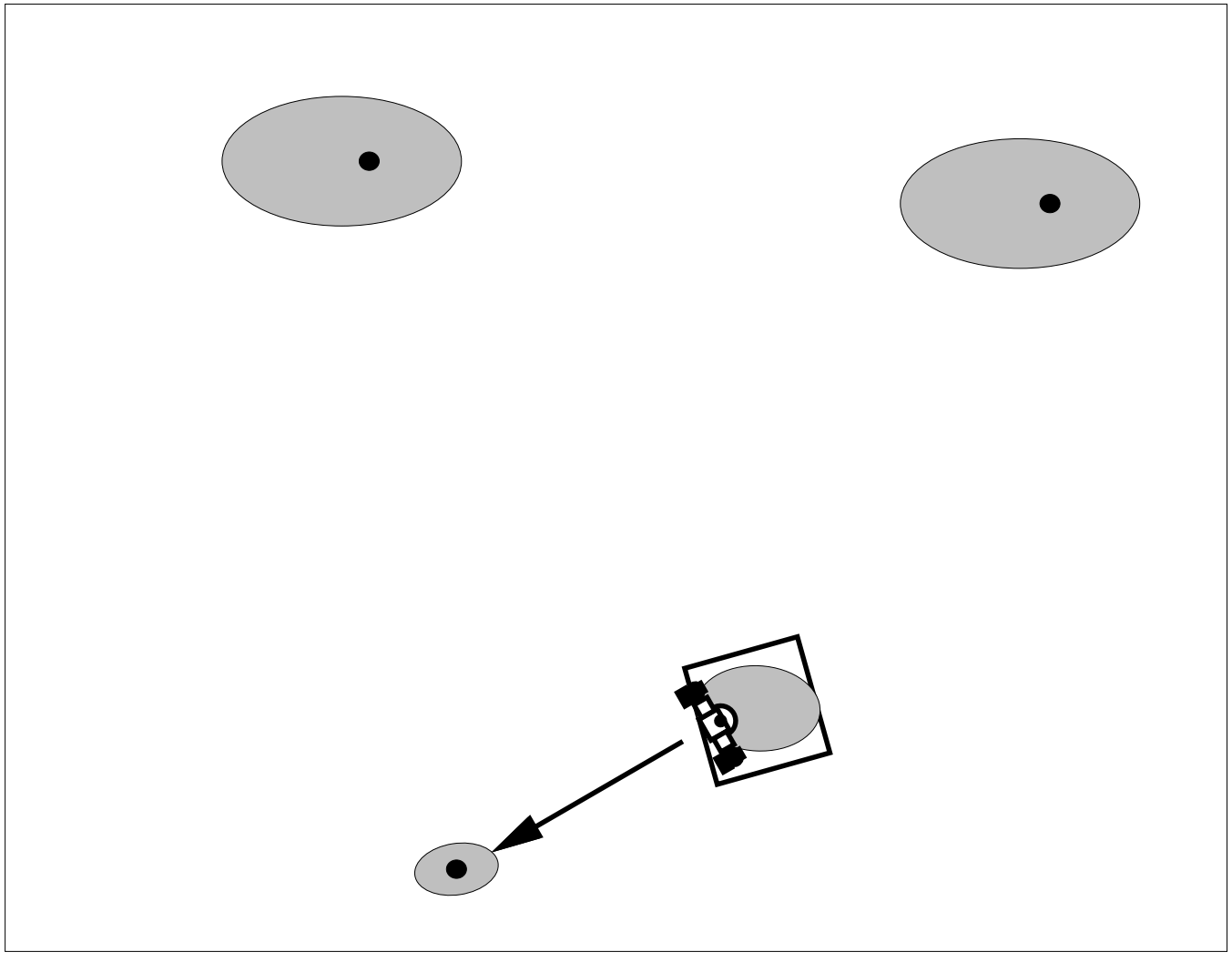
(c) Robot makes first measurements of B and C.

Simultaneous Localisation and Mapping



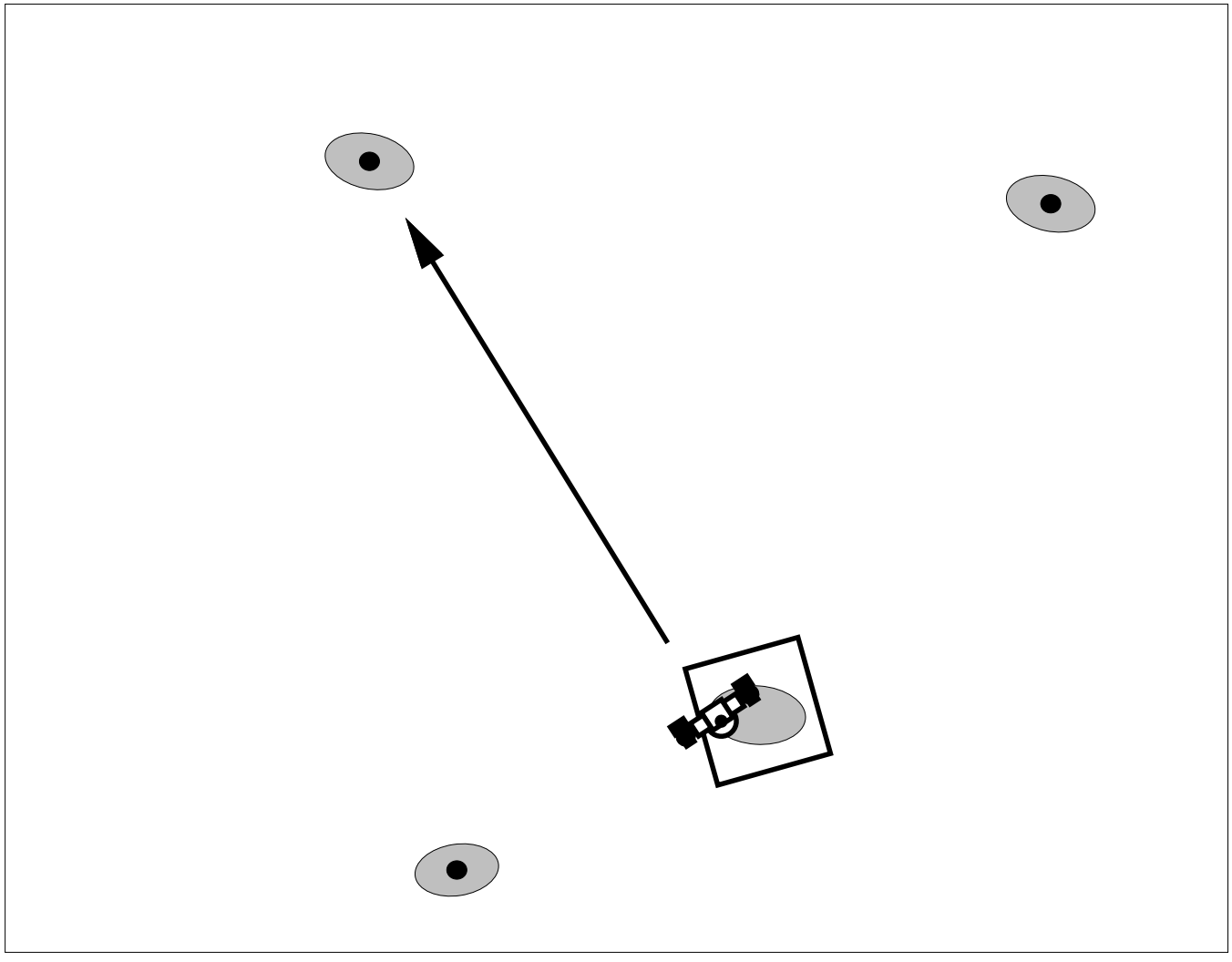
(d) Robot drives back towards start (uncertainty grows more)

Simultaneous Localisation and Mapping



(e) Robot re-measures A; *loop closure*! Uncertainty shrinks.

Simultaneous Localisation and Mapping



(f) Robot re-measures B; note that uncertainty of C also shrinks.

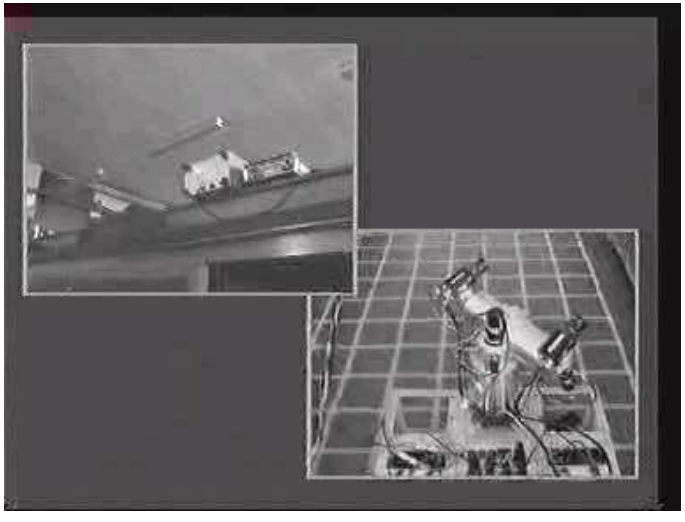
SLAM with First Order Uncertainy Propagation

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{\mathbf{x}}_v \\ \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \\ \vdots \end{pmatrix} \quad , \quad \mathrm{P} = \begin{bmatrix} \mathrm{P}_{xx} & \mathrm{P}_{xy_1} & \mathrm{P}_{xy_2} & \cdots \\ \mathrm{P}_{y_1x} & \mathrm{P}_{y_1y_1} & \mathrm{P}_{y_1y_2} & \cdots \\ \mathrm{P}_{y_2x} & \mathrm{P}_{y_2y_1} & \mathrm{P}_{y_2y_2} & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix}$$

- Camera pose and map stored in single state vector and updated on every frame via a single Extended Kalman Filter.
- Full PDF over robot and map parameters represented by a single multi-variate Gaussian.

SLAM Using Vision: First Steps

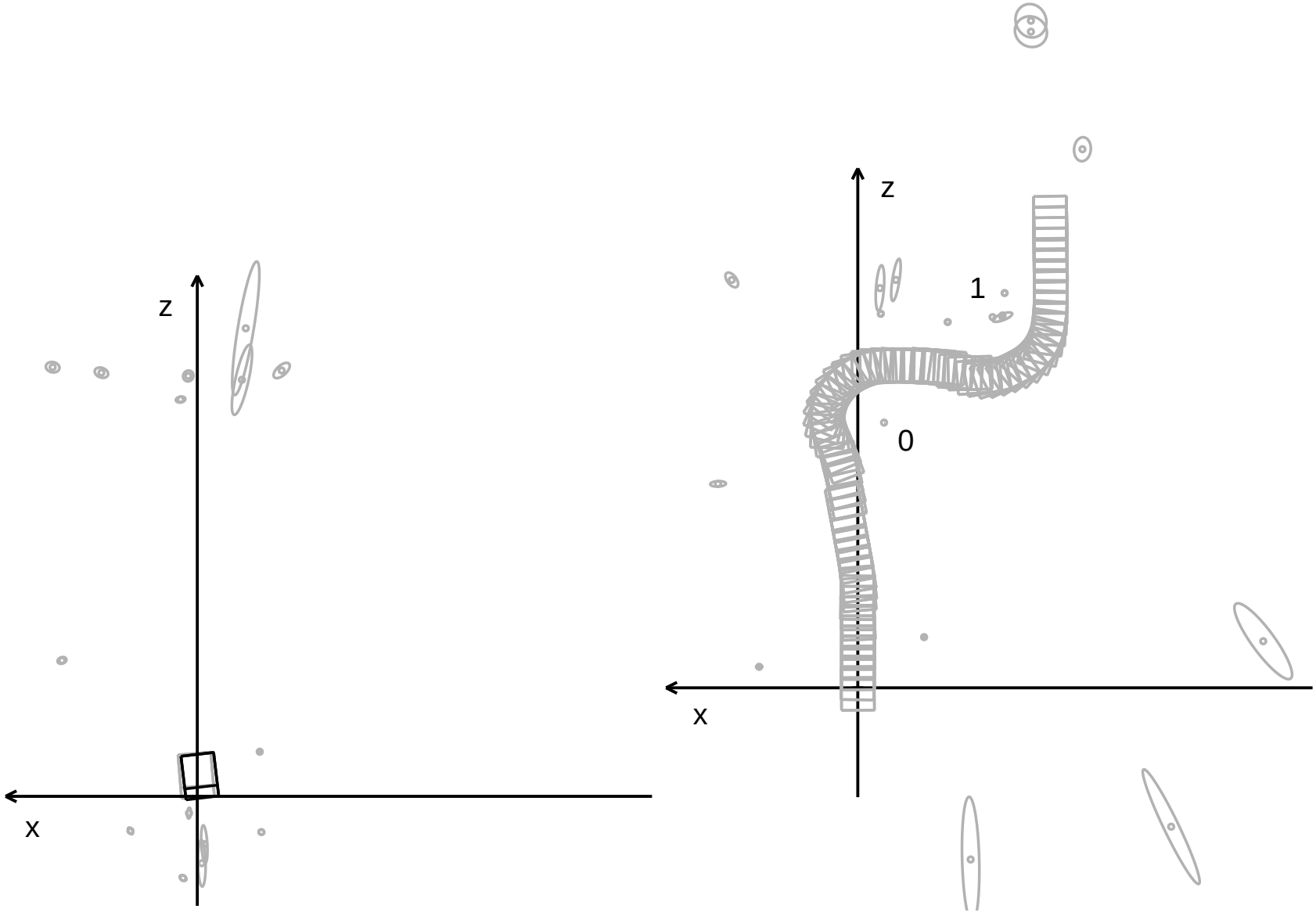
- Fixating active stereo measuring one feature at a time.
- 5Hz real-time processing (100MHz PC!).



Davison and Murray, ECCV 1998, PAMI 2002.

SLAM Using Active Stereo Vision

Probabilistic Map Results

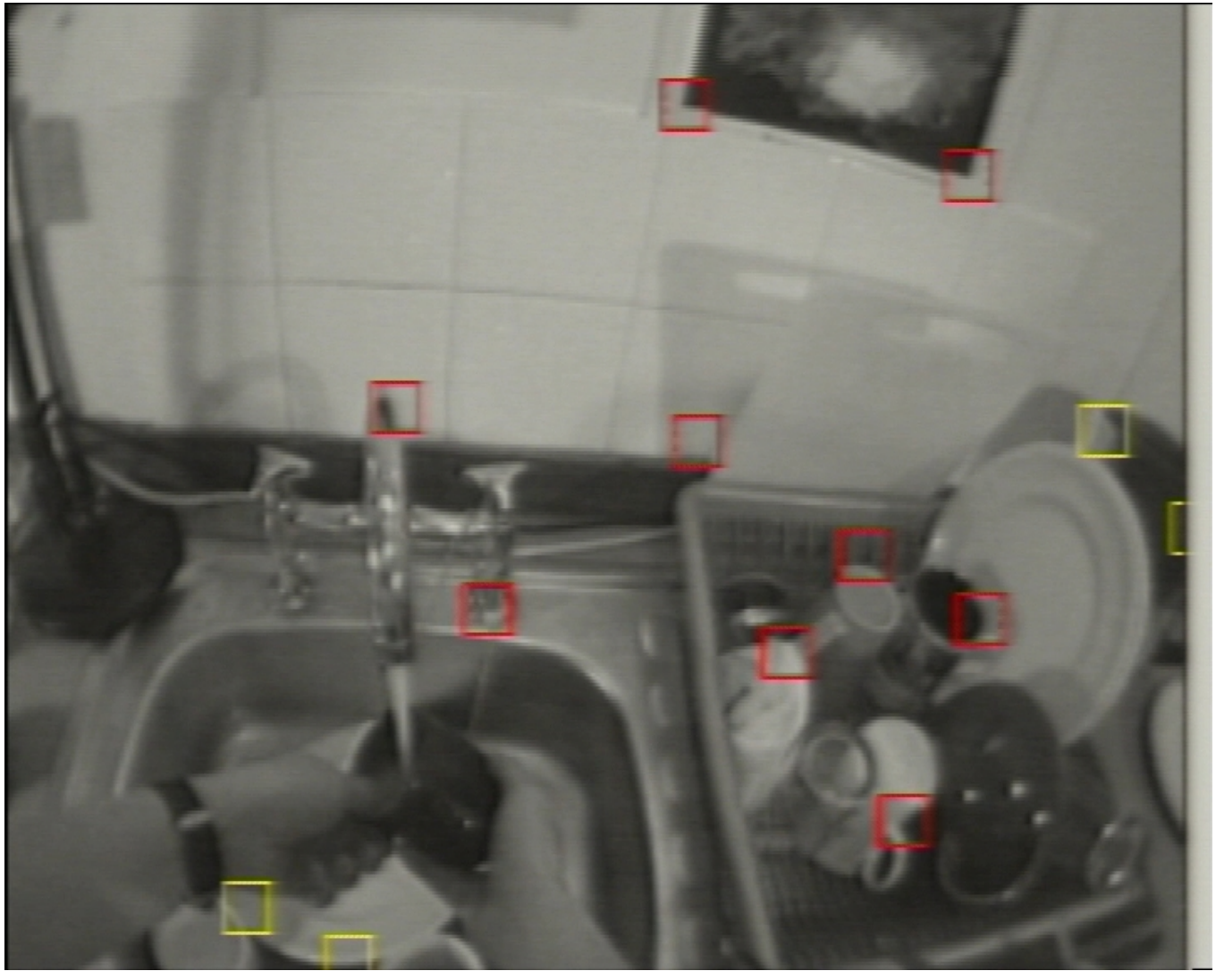


Monocular SLAM

- Can we still do SLAM with a single unconstrained camera, flying generally through the world in 3D, hand-held or carried by a robot or person?

- Aim to build, initially for a local area, a persistent map which enables *drift-free localisation*.
- Can monocular SLAM gradually evolve into a general real-time 3D perception and scene understanding capability?
- Always sequential, model-based and taking account of physics; but in general trying to avoid domain-specific assumptions.
- Start from what we can do in real-time with a single camera and build it up from there, riding rising computer power and better algorithms.

MonoSLAM

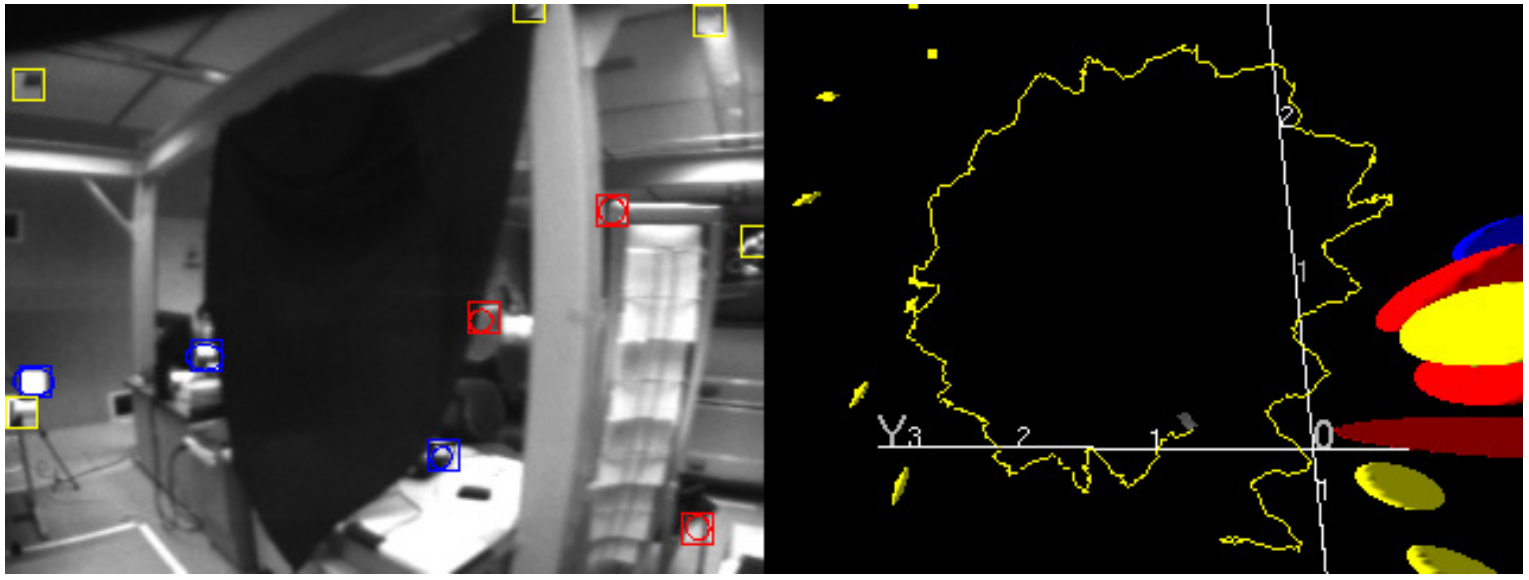


Davison, ICCV 2003; Davison, Molton, Reid, Stasse, PAMI 2007.

Application: HRP-2 Humanoid at JRL, AIST, Japan

- Small circular loop within a large room
- No re-observation of 'old' features until closing of large loop.

HRP2 Loop Closure

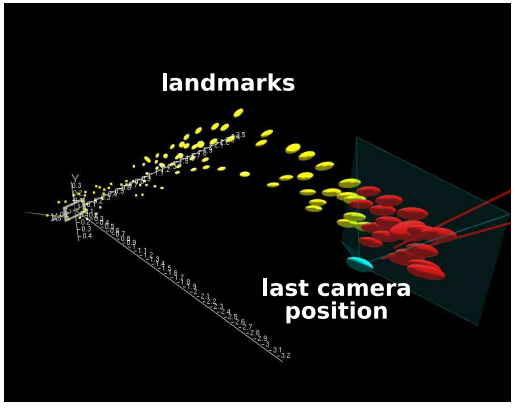


(Davison, Stasse, *et al.*, PAMI 2007)

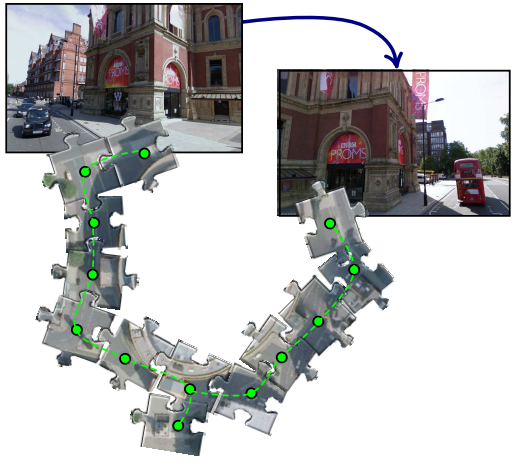
Application: Wearable Robot SLAM

- Davison, Mayol and Murray, ISMAR 2003.

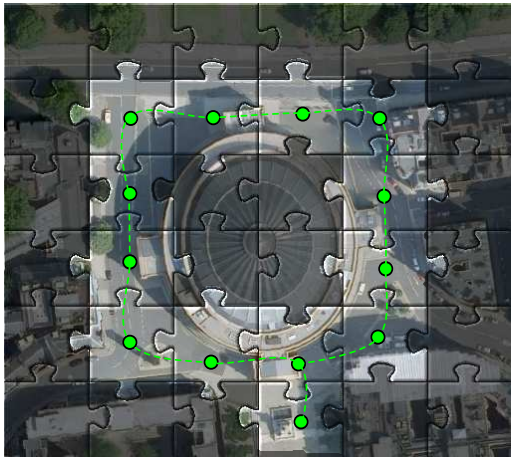
General Components of a Scalable SLAM System



Local Motion Estimation

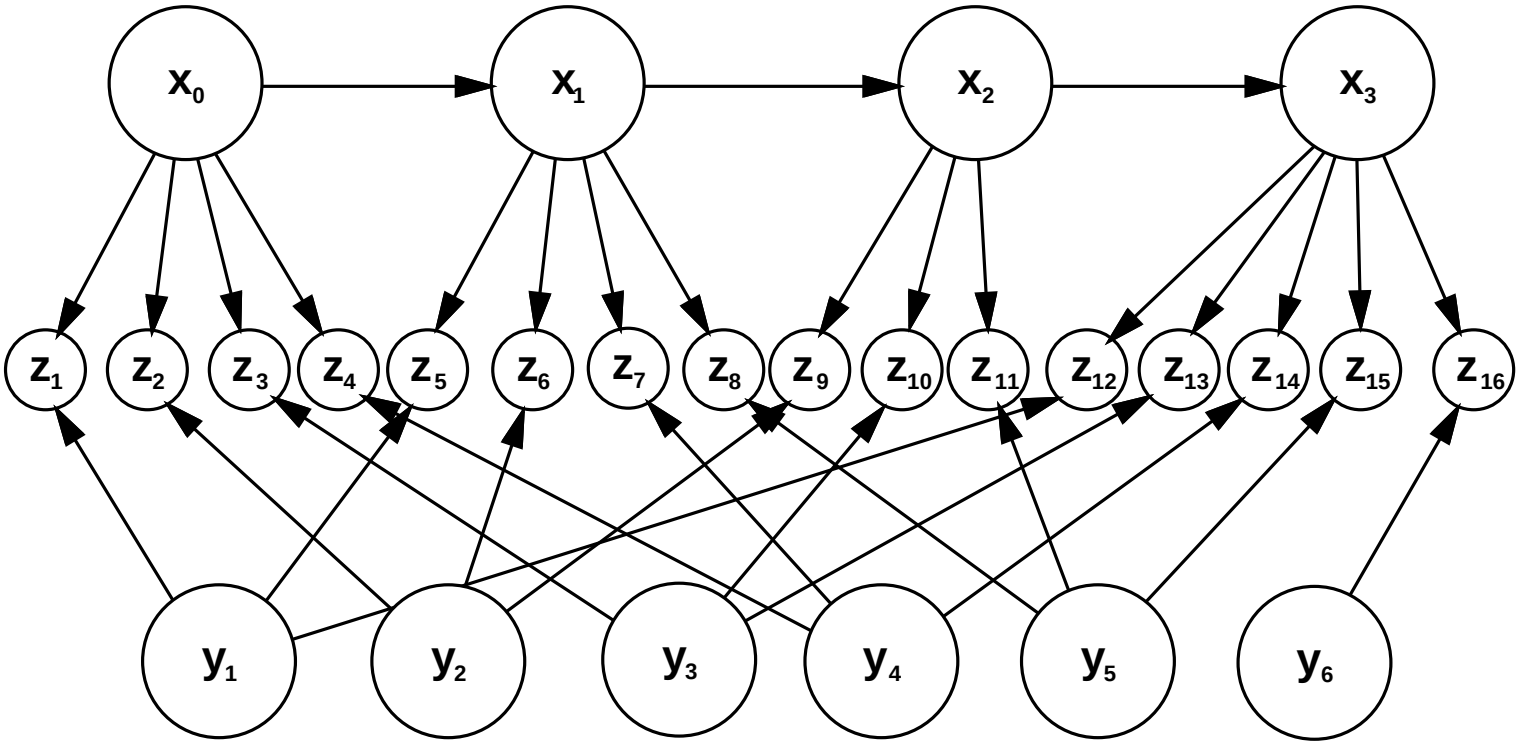


Loop Closure Detection



Global Map Relaxation

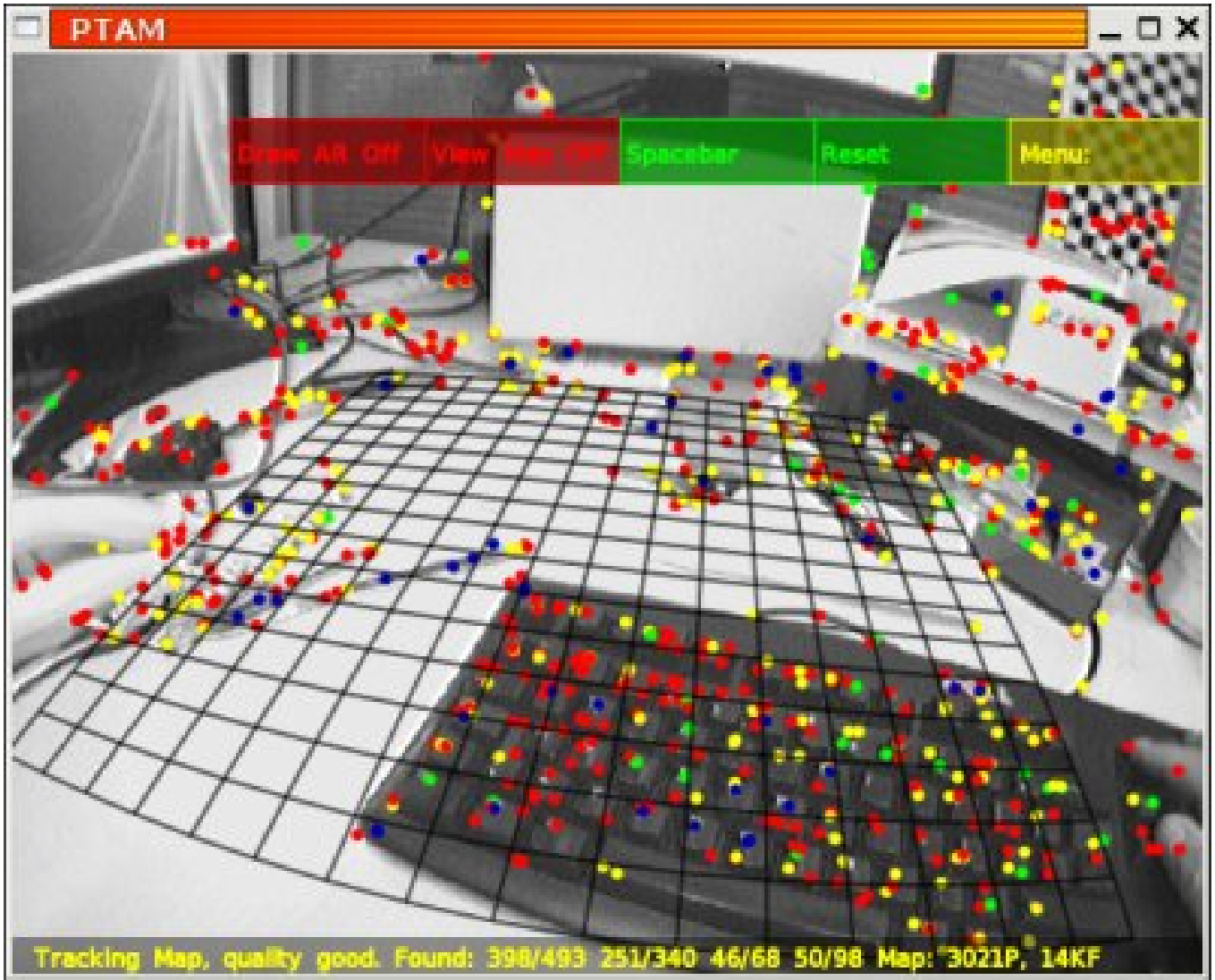
SLAM as a Bayesian Network



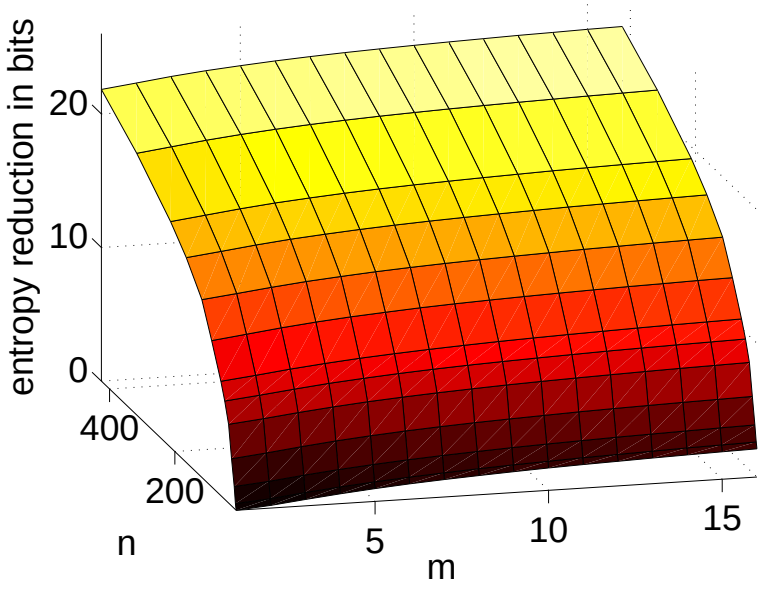
(See e.g. 'Probabilistic Robotics', Thrun, Burgard and Fox, MIT Press 2005, or much work by Dellaert, Konolige and others.)

PTAM: Parallel Tracking and Mapping

2007,2008 Klein and Murray's PTAM (ISMAR 2007), also passive, optimised software using features of the CPU. Maps are much denser than MonoSLAM.

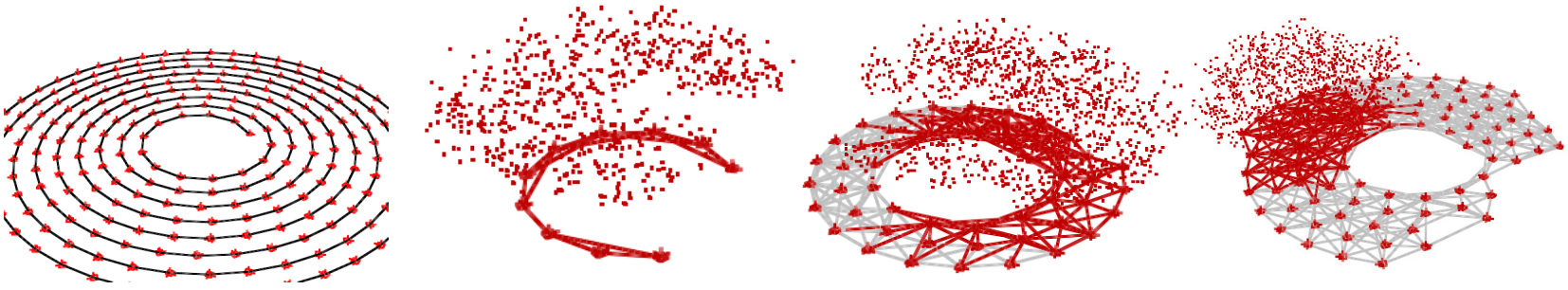


Real-Time Monocular SLAM: Why Filter?



- Hauke Strasdat, J. M. M. Montiel and Andrew J. Davison, ICRA 2010.
- A comparison: filtering vs. keyframes + optimisation for monocular SLAM in terms of accuracy and computational cost.
- A clear winner with modern computing resources: keyframes + optimisation.

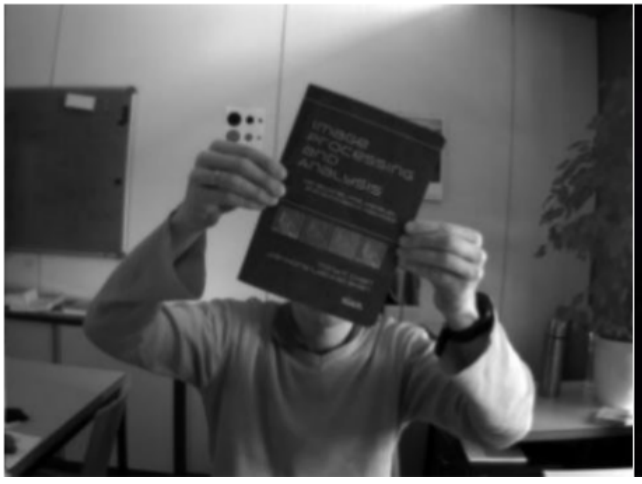
Double Window Optimisation for Constant Time Visual SLAM



Spiral | Inner window | Inner & outer | Constant time

- Simultaneously optimises in full BA style in an inner window, and around a pose graph in an outer window.
- Applied to monocular and stereo vision for either loopy or exploratory journies; also RGBD.
- Hauke Strasdat, Kurt Konolige, José María Montiel and Andrew Davison, ICCV 2011.

Towards Live Dense Reconstruction



- Can we go beyond point feature-based reconstructions in real-time?
- Dense optical flow and multi-view stereo literature very developed in computer vision (very closely linked problems).
- TV-L1 Optical Flow on the GPU (Zach, Pock, Bischof, TU Graz, DAGM 2007); `www.gpu4vision.org`.

Solving Inverse Problems with Variational Optimisation

- TV-L1 energy for image denoising.

$$E(u) = \int_\Omega |\nabla u| \mathrm{d}\Omega + \lambda \int_\Omega |I_0(\mathbf{x}) - u(\mathbf{x})| \mathrm{d}\Omega$$

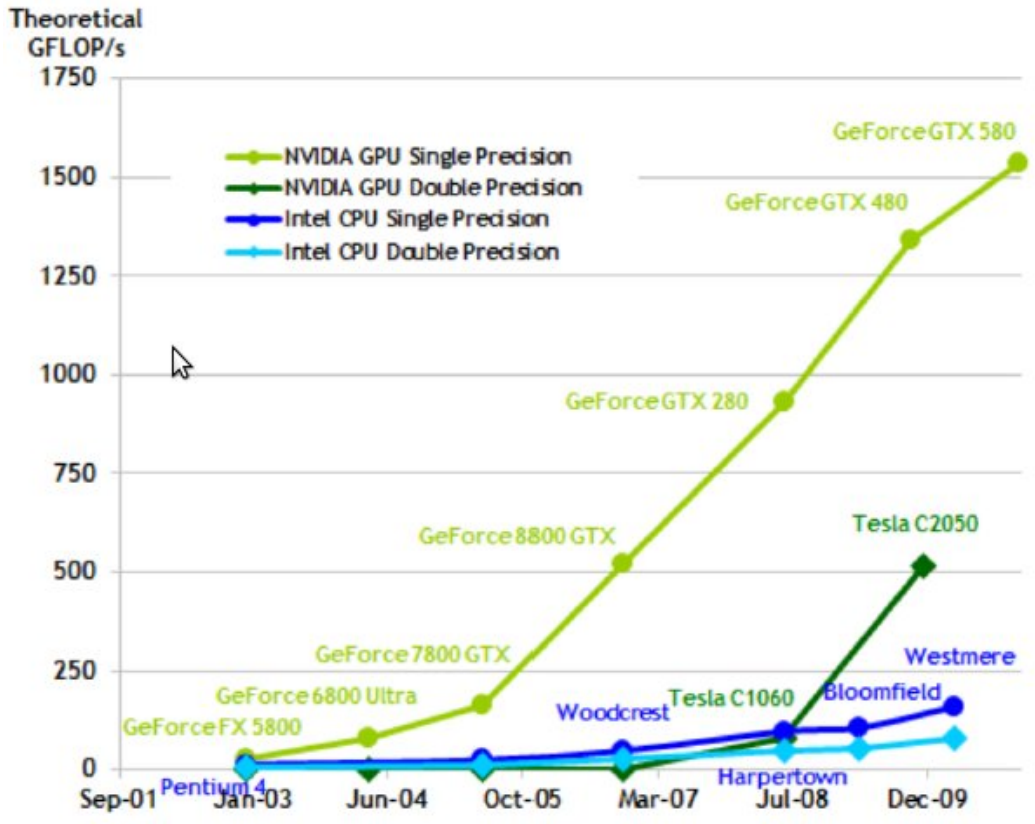
Original Noise Added Denoised: L2 Denoised: L1

- TV-L1 energy for optical flow estimation.

$$E(\mathbf{u}) = \int_\Omega |\nabla \mathbf{u}| \mathrm{d}\Omega + \lambda \int_\Omega |I_0(\mathbf{x}) - I_1(\mathbf{x} + \mathbf{u}(\mathbf{x}))| \mathrm{d}\Omega$$

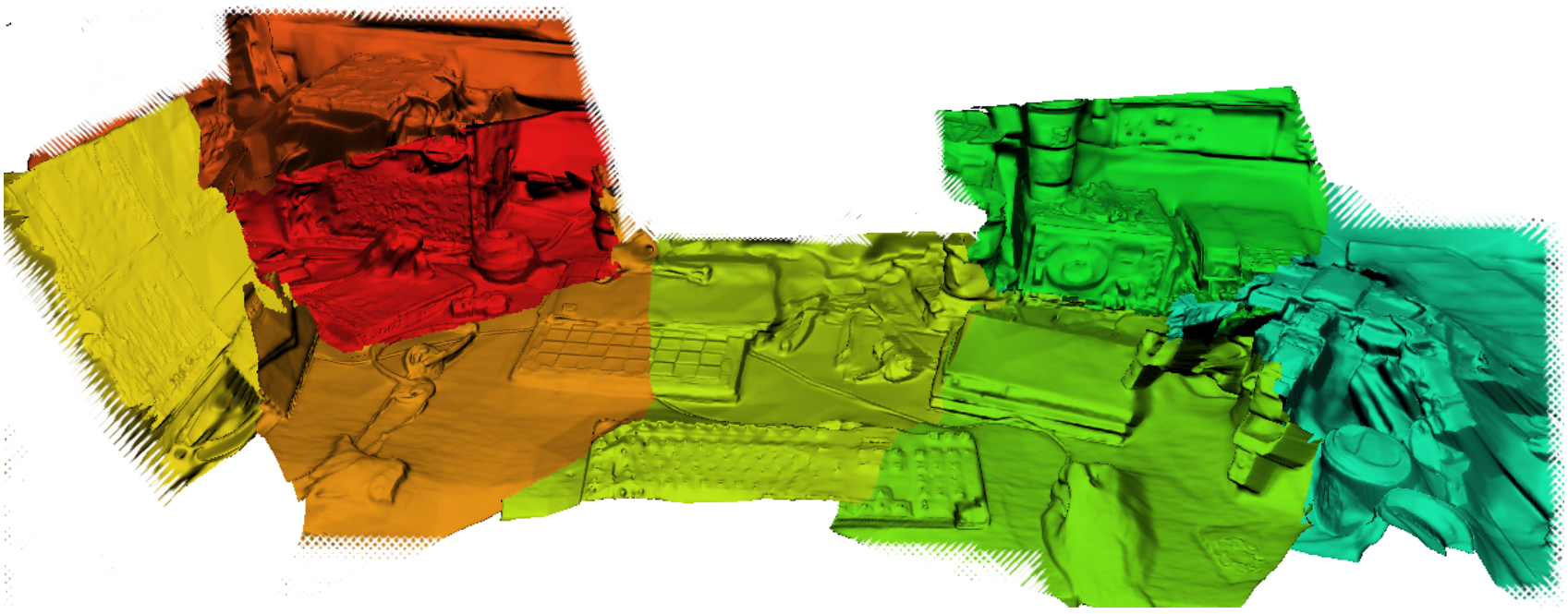
(See PhD thesis of Thomas Pock, TU Graz, for a great introduction.)

GPGPU Processing Power



- Massively parallel processing has taken over as the dominant current computing paradigm (and it's hard to see it crossing back over).
- Current gaming GPU: NVidia GTX 580, 512 CUDA cores, up to 3GB RAM, $500.

Live Dense Reconstruction with a Single Camera



(Newcombe, Davison, CVPR 2010)

- During live camera tracking (point-based real-time monocular PTAM), select small bundles of frames for dense depth map creation on GPU via view-predictive optical flow.
- Each depth map turned into a mesh of 640 $\times$ 480 vertices; multiple depth maps put side by side but could be fused.
- Live operation on current desktop/laptop hardware.

Spherical Mosaicing using Whole Image Alignment

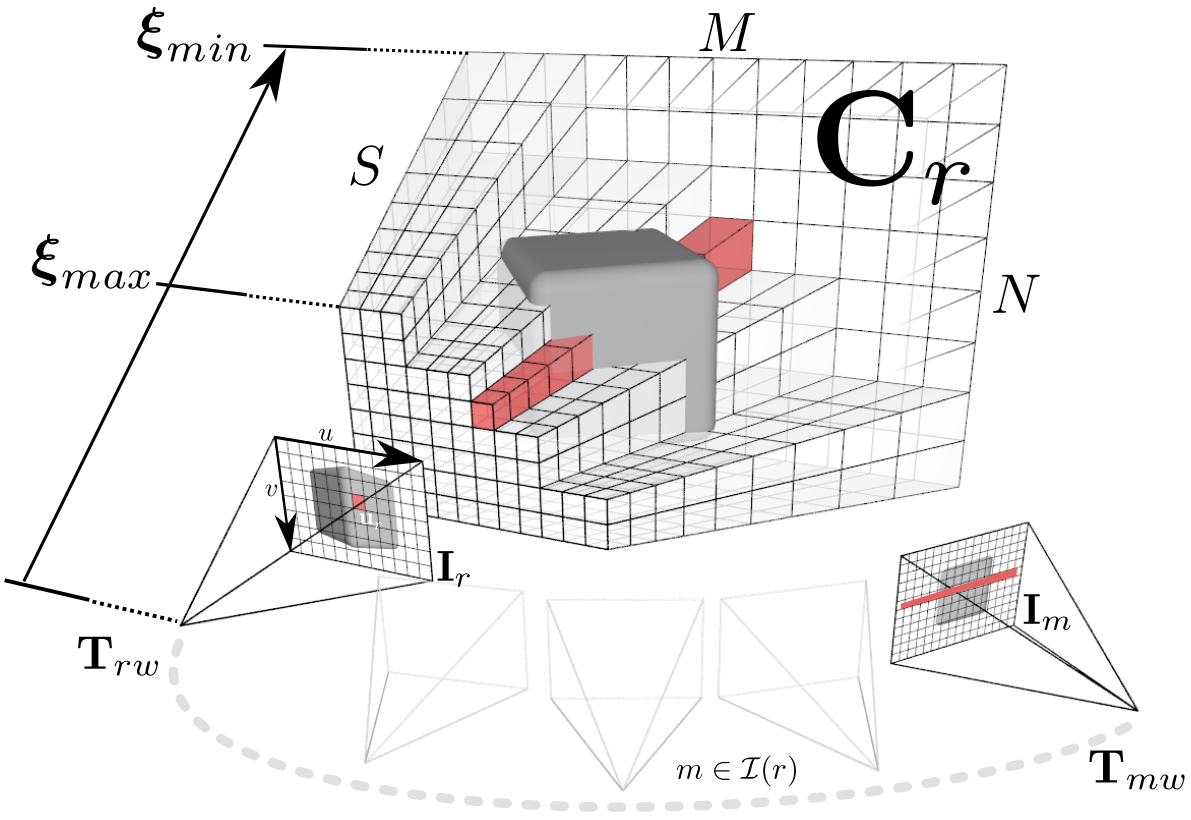
- Keyframe-based spherical mosaicing, Lovegrove and Davison, ECCV 2010.
- Tracking by whole image alignment on a pyramid; GPU implementation.
- Whole image alignment tracking robust to fast motion, blur.
- Interleaved global optimisation of keyframe set and camera intrinsics.

DTAM: Dense Tracking *and* Mapping

- Dense Mapping: Directly solve for depth maps using variational optimisation 100s of small baseline images, given known camera trajectory.
- Dense Tracking Close the tracking and mapping loop by tracking the camera pose against the current dense surface prediction moving, away from sparse features and point clouds altogether.
- Newcombe, Lovegrove, Davison, ICCV 2011.

Cost volume data term

Build a cost volume from lots of weak data terms, and then using a simple discontinuity preserving smoothness prior, optimise global energy.



Using all possible frames from the live camera

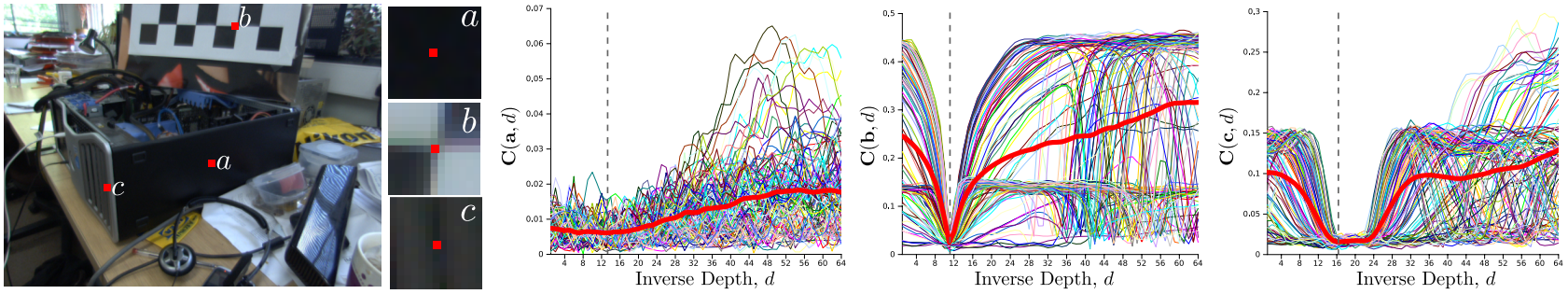


Figure: Plots for the single pixel photometric functions (absolute differences of RGB values) and the sum across multiple images (shown in thick red line).

Figure: Per pixel inverse depth minimum for increasing numbers of data terms (shown in left three), in comparison to the sparse points found by binary data-association methods in PTAM.

Tracking using the dense model

The dense surface prediction enables a simple way to perform camera tracking using all possible pixels in the live image: $\mathbb{SE}(3)$ pose esimtaiton using a $2.5D$ Lucas-Kanade style optimisation with a per pixel data error:

$$f_{\mathbf{u}}(\boldsymbol{\psi}) = \mathbf{I}_l \left(\pi \left(\mathtt{KT}_{lv}(\boldsymbol{\psi}) \pi^{-1} \left(\mathbf{u}, \xi_v \left(\mathbf{u} \right) \right) \right) \right) - \mathbf{I}_v \left(\mathbf{u} \right).$$

This uses a predicted vertex map into the known previous frame, and a predicted RGB image in the same frame using OpenGL for rendering.

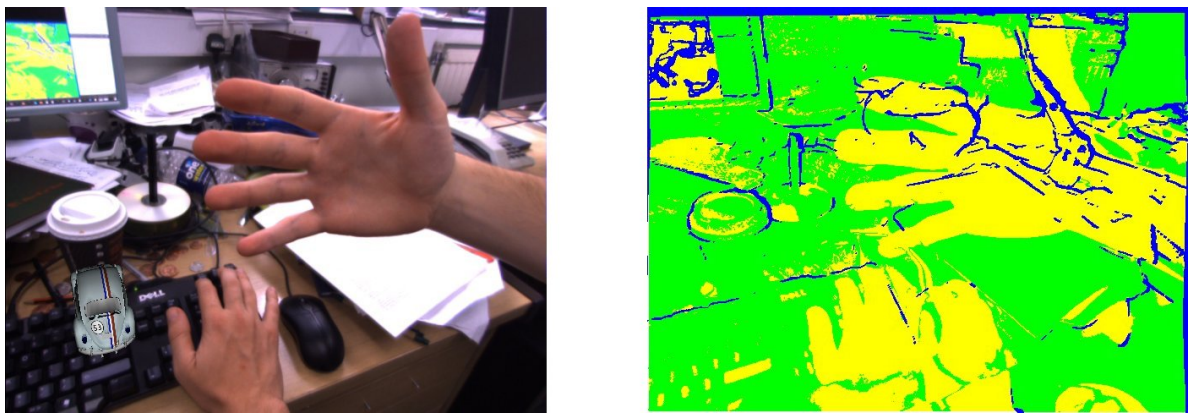
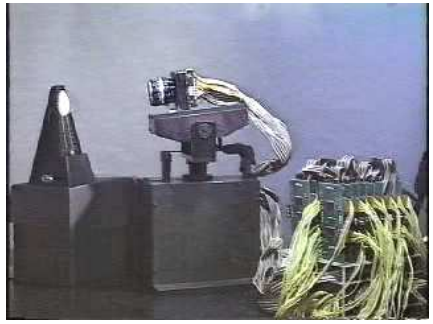


Figure: Gating given the predicted and live image (shown left).

Why Not Increase Frame-Rate to Track Even Faster Motion?

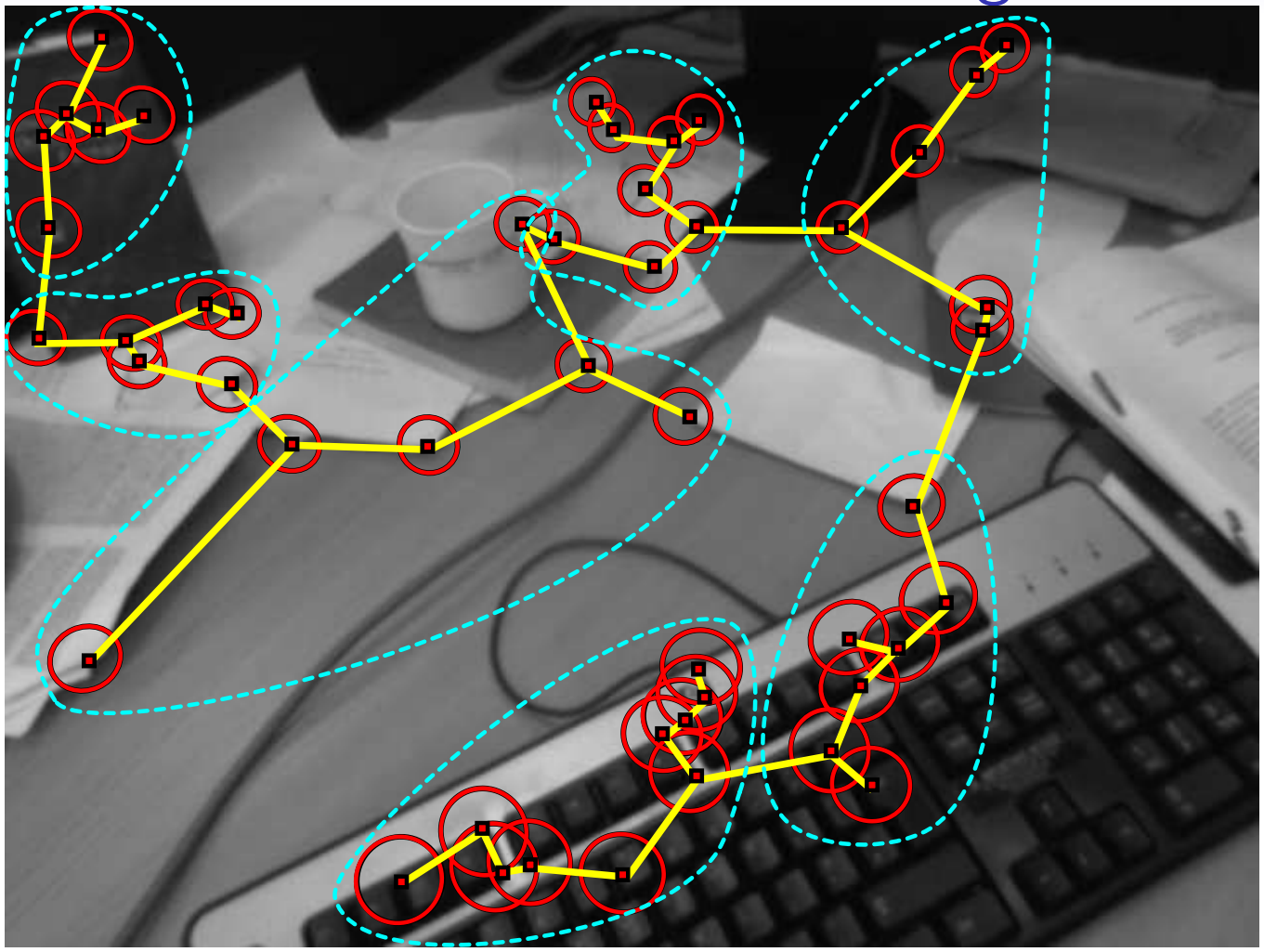


Real-time tracking

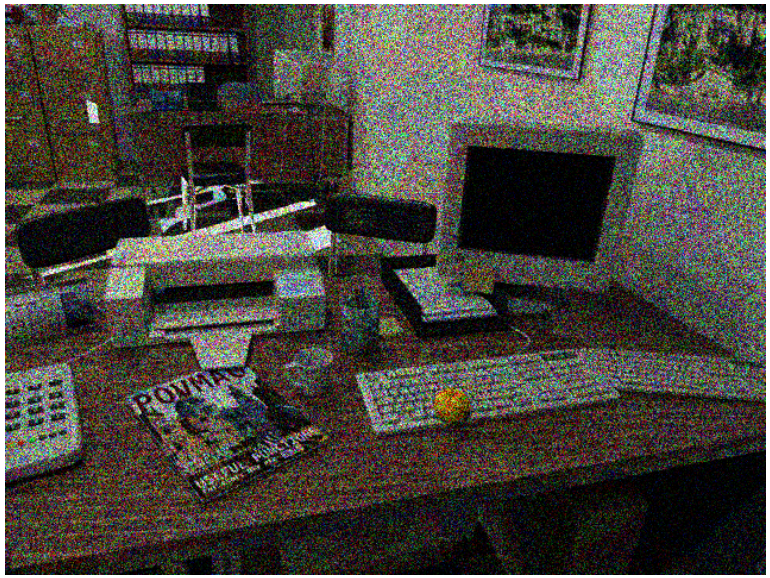
- High frame-rate seems better but.. today most advanced real-time tracking is at 10–60Hz.
- Why? Should we increase the frame-rate in real, modern advanced tracking problems?
- In trackers that benefit from prediction, computational cost *per frame* decreases as frame-rate increases.

Scalable Active Matching



- Efficient transfer of matching result from feature to feature by message passing through a tree.
- Handa, Chli, Strasdat, Davison, CVPR 2010.

Experimental Investigation with 'Photo-Realistic' Video Generated from Ray Tracing with Realistic Noise and Blur



100Hz video

20Hz video

- Analyse DTAM-style whole image alignment.
- Input space: frame-rate and resolution.
- Multi-objective evaluation criteria: accuracy, robustness and computational cost.
- Results depend greatly on scene lighting level.
- Handa, Newcombe, Angeli and Davison, ECCV 2012.

Experiment assuming perfect lighting

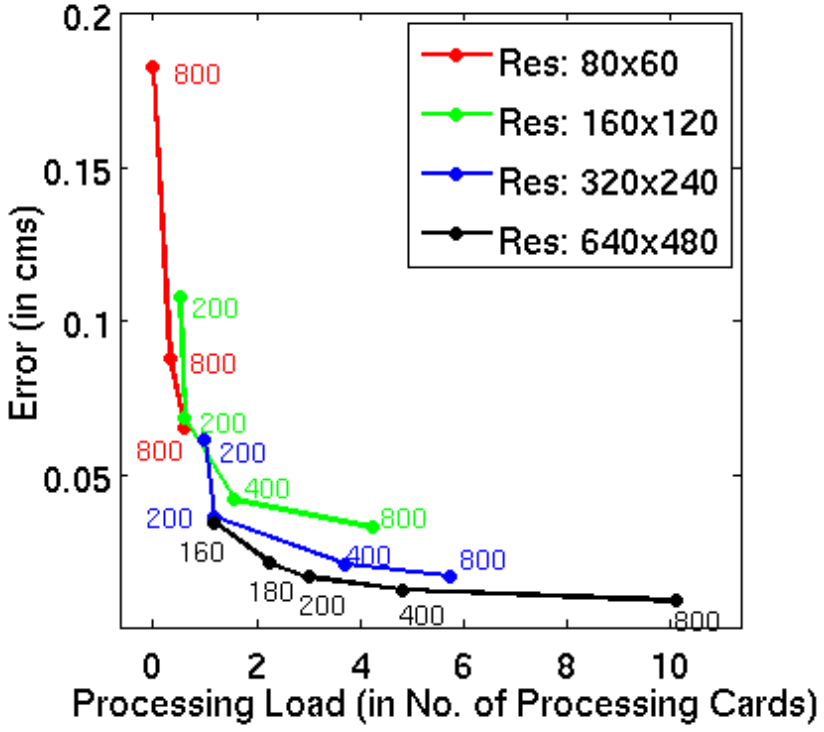
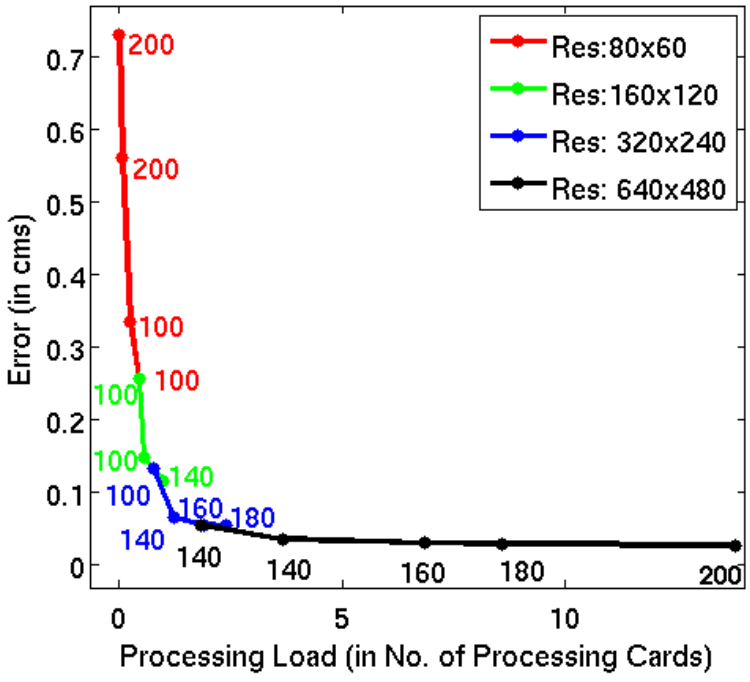


Figure: Pareto front for mininum error/minimum processing load performance, highlighting with numbers the frame-rates that are optimal for each available budget.

Interpretations

- No noise and no blur — perfect lighting conditions.
- For very low budget few iterations on higher frame-rates (800Hz) are sufficient because baseline is already small to achieve the accuracy.
- Crossovers as the budget changes.
- A combination of high frame-rate and high resolution works best as budget increases.

Moderate lighting



α=10

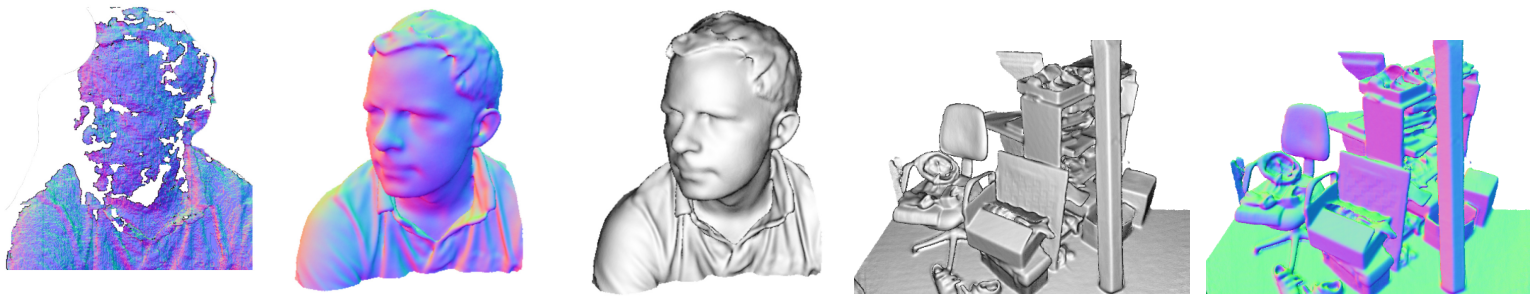
Figure: Pareto Fronts for lighting level $\alpha = 10$.

Interpretations for α=10, moderate lighting

- 200Hz is best choice for very very low budget because prediction is strong and few iterations are sufficient.
- A slight increases sees 100Hz as the best choice — contrast to high as well as perfect lighting conditions.
- Best choice of frame-rates shift to slightly lower values compared to high lighting.

Real-Time Dense Surface Mapping and Tracking

KinectFusion (ISMAR 2011, Newcombe with Izadi *et al.*at Microsoft Research Cambridge). We fuse 30Hz depth maps from Kinect into a global implicit surface.



Builds a fused, always up-to-date volumetric scene model

For the first time, we then track the current frame against the complete fused model massiveley improving tracking ability with surprising results for global consistency. We use only depth data and the implementation is designed to exploit GPGPU.

Signed Distance Function surface representations

We use a *truncated signed distance* function representation, $\mathrm{F}(\vec{x}) : \mathbb{R}^3 \mapsto \mathbb{R}$ for the estimated surface where $\mathrm{F}(\vec{x}) = 0$.

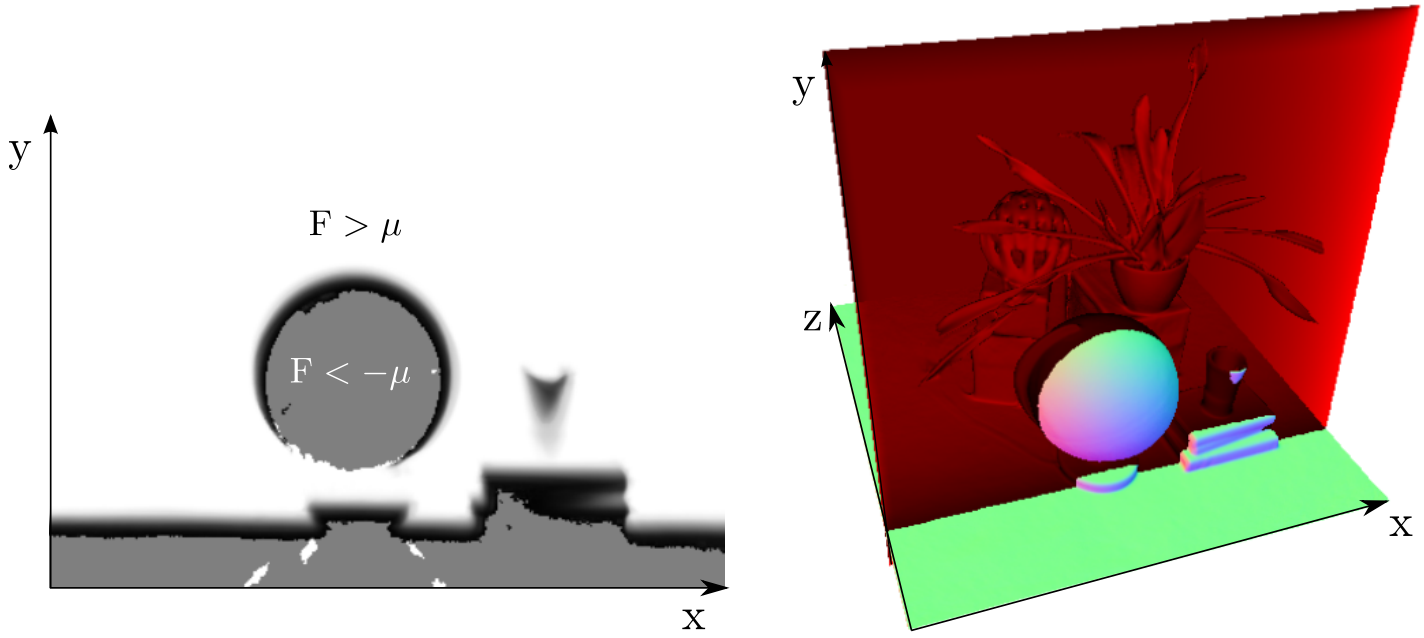
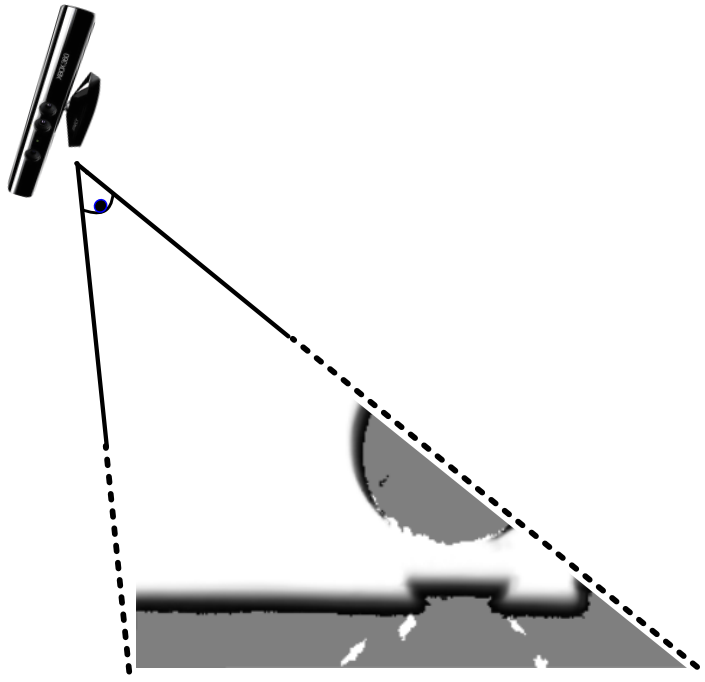
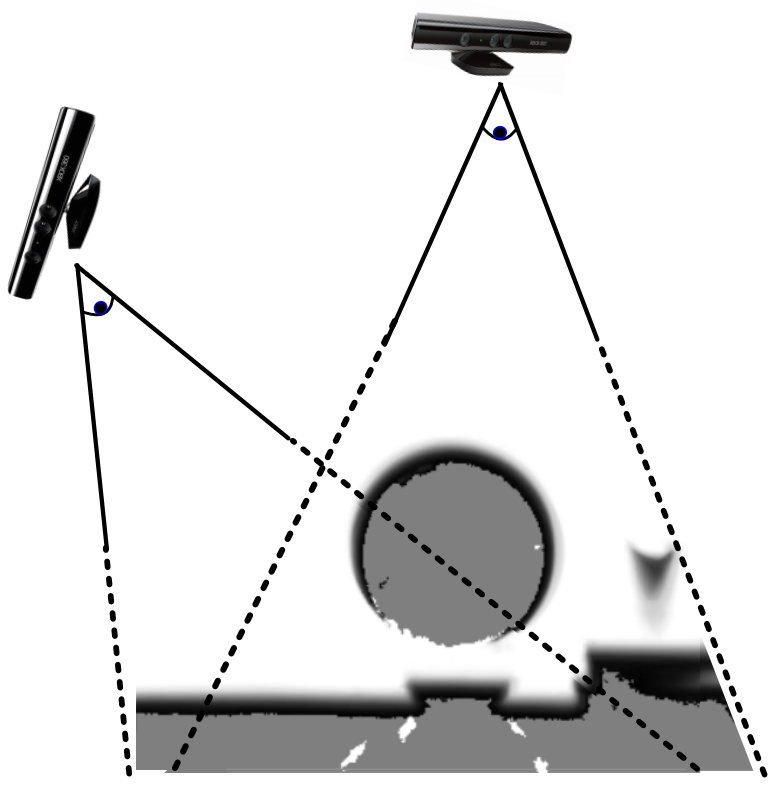


Figure: A cross section through a $3D$ Signed Distance Function of the surface shown.

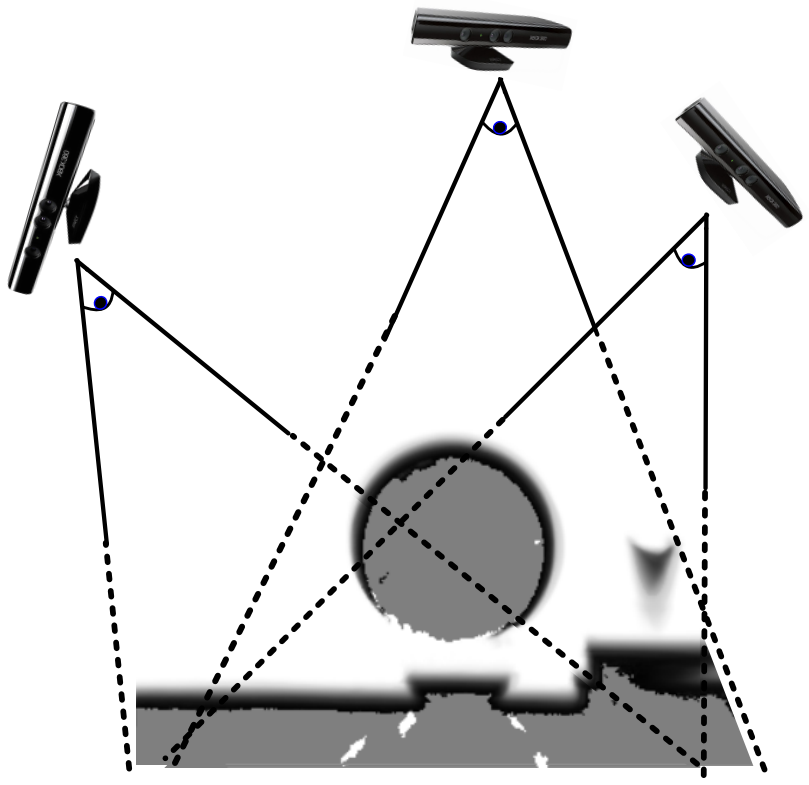
SDF Fusion



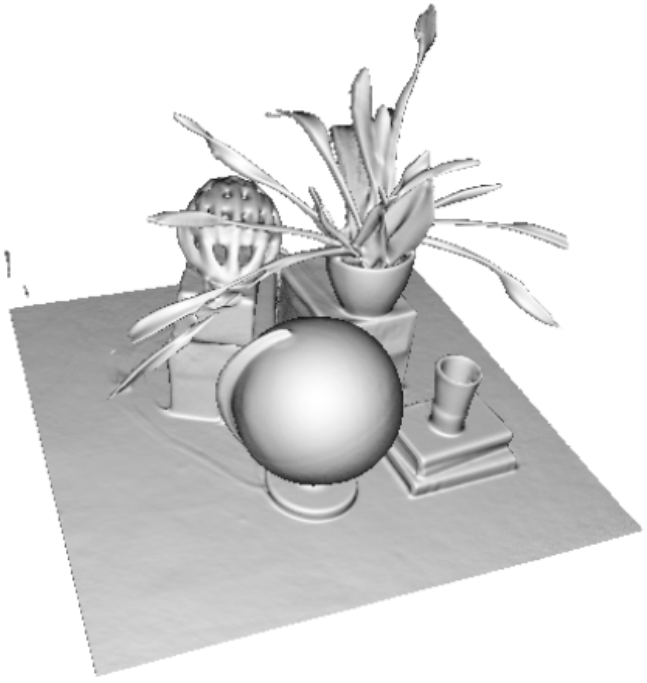
SDF Fusion



SDF Fusion

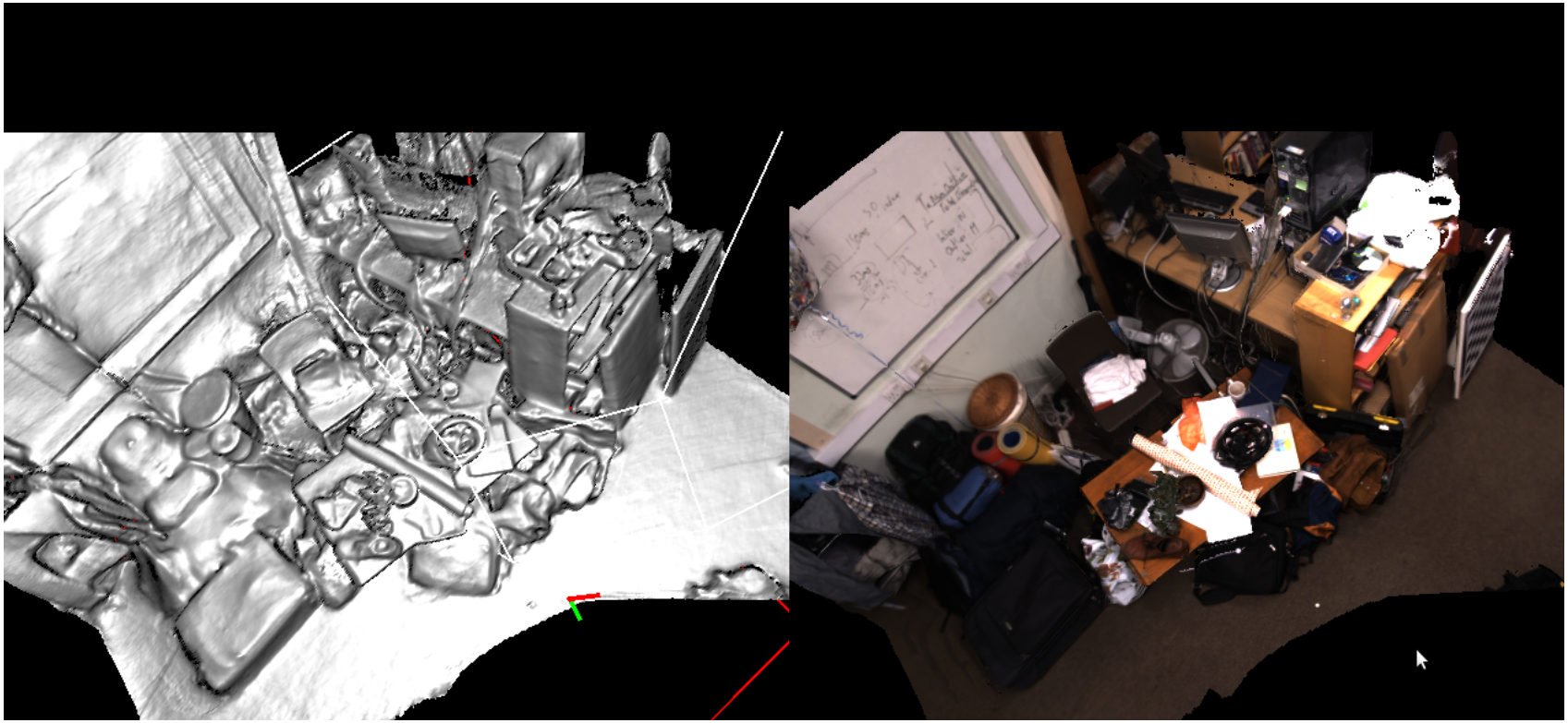


SDF Fusion



Similar to volumetric denoising of the SDF under an $\mathcal{L}_2$ norm data-cost with no regularisation: Can be computed online as data comes in using weighted average.

Real-time Surface Fusion using a Single RGB Camera



- Richard Newcombe, 2011-2012.
- This result demonstrates the dramatic improvements in real-time SLAM using commodity hardware over the past decade; the same single camera input as MonoSLAM used and the computing hardware costs about the same.

Towards Greater Physical Understanding: Surface Light-Field Capture for Reflectance and Lighting Estimation

- Specular reflections are currently at best ignored by tracking and reconstruction systems; but should be a valuable source of information for both tracking and detailed reconstruction.
- First steps: real-time surface light-field capture, and AR for planar specular surfaces.
- Jachnik, Newcombe and Davison, ISMAR 2012.