

Robust Real-time Visual Monocular SLAM

**Denis Chekhlov, Mark Pupilli,
Walterio Mayol and Andrew Calway**

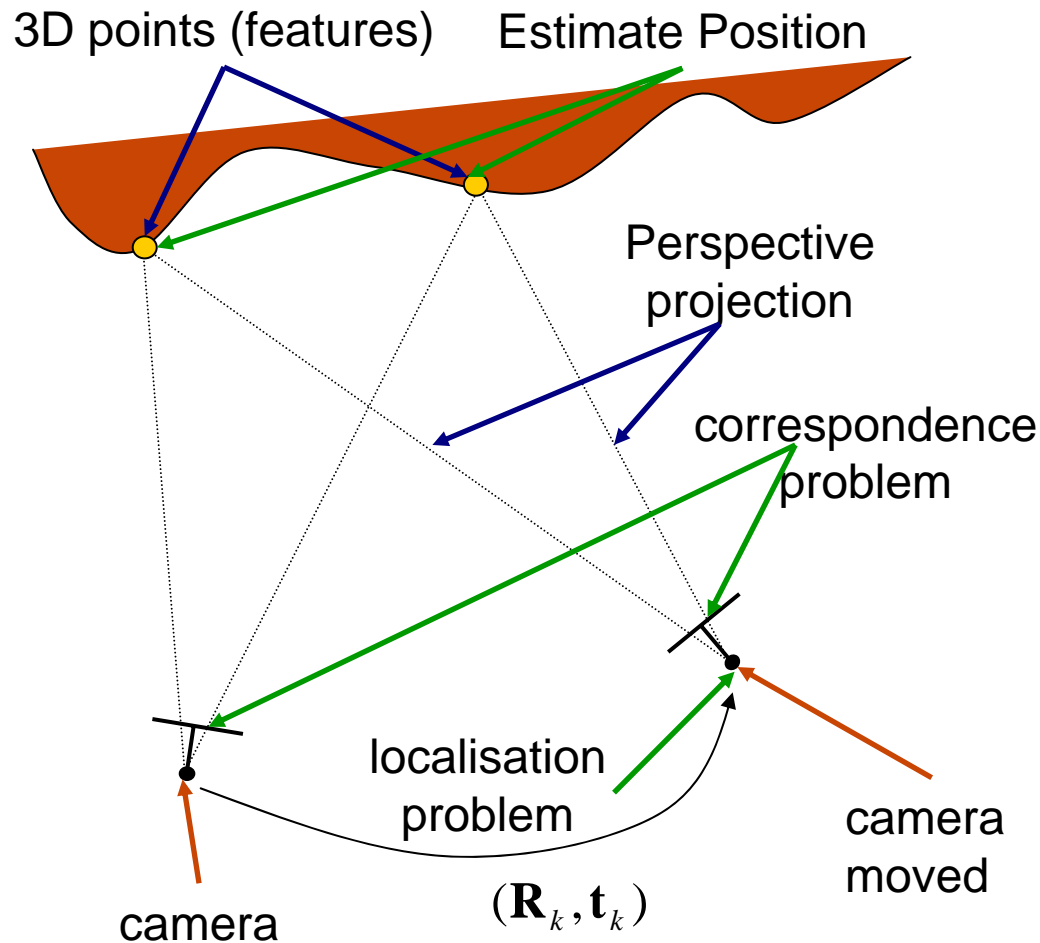
**Department of Computer Science
University of Bristol, UK**



14 December 2006

Introduction to visual SLAM

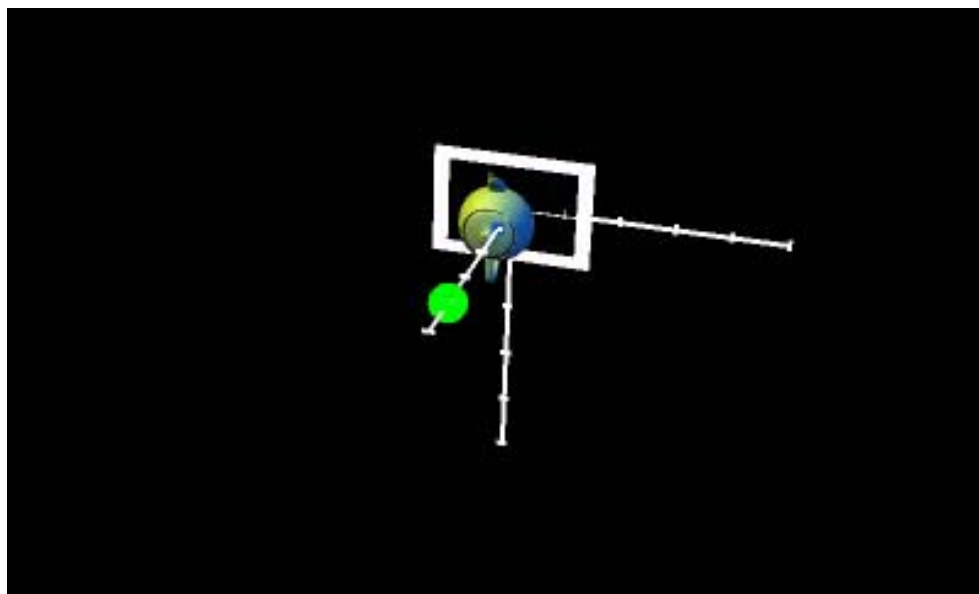
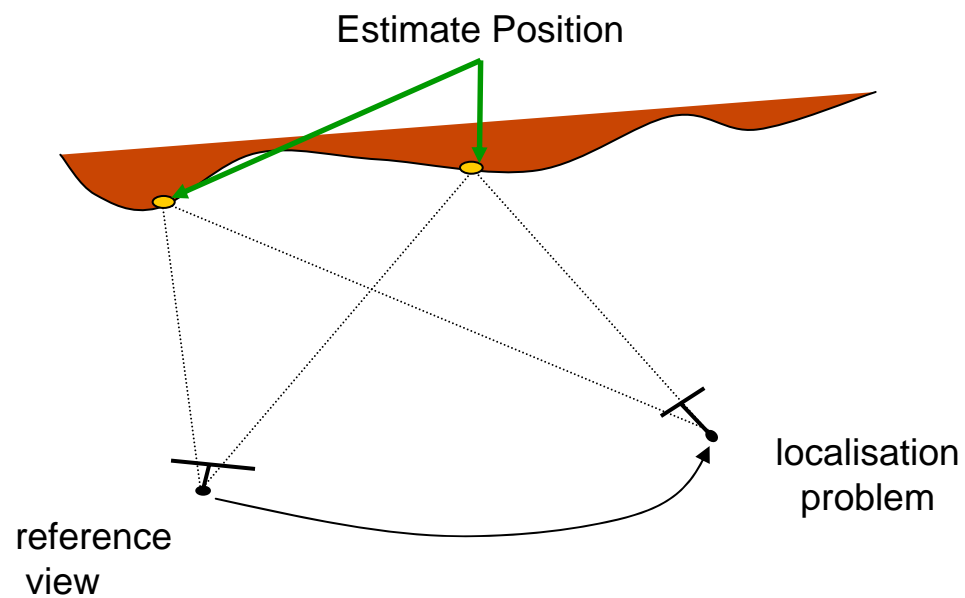
SLAM – Simultaneous *L*ocalization *A*nd *M*apping



We aim to

- **Localise camera** (6DOF – Rotation and Translation from reference view)
- While **simultaneously** estimate **3D map** of features (3D points)

Introduction to visual SLAM



Visual SLAM is a challenging problem:

- Needs to be run in real-time (< 40 ms \sim 25 frames per second)
- Correspondence problem – identification of 3D points in each frame is really hard
- Position of the camera and location of 3D points are uncertain
- These uncertainties must be estimated to achieve reliable performance

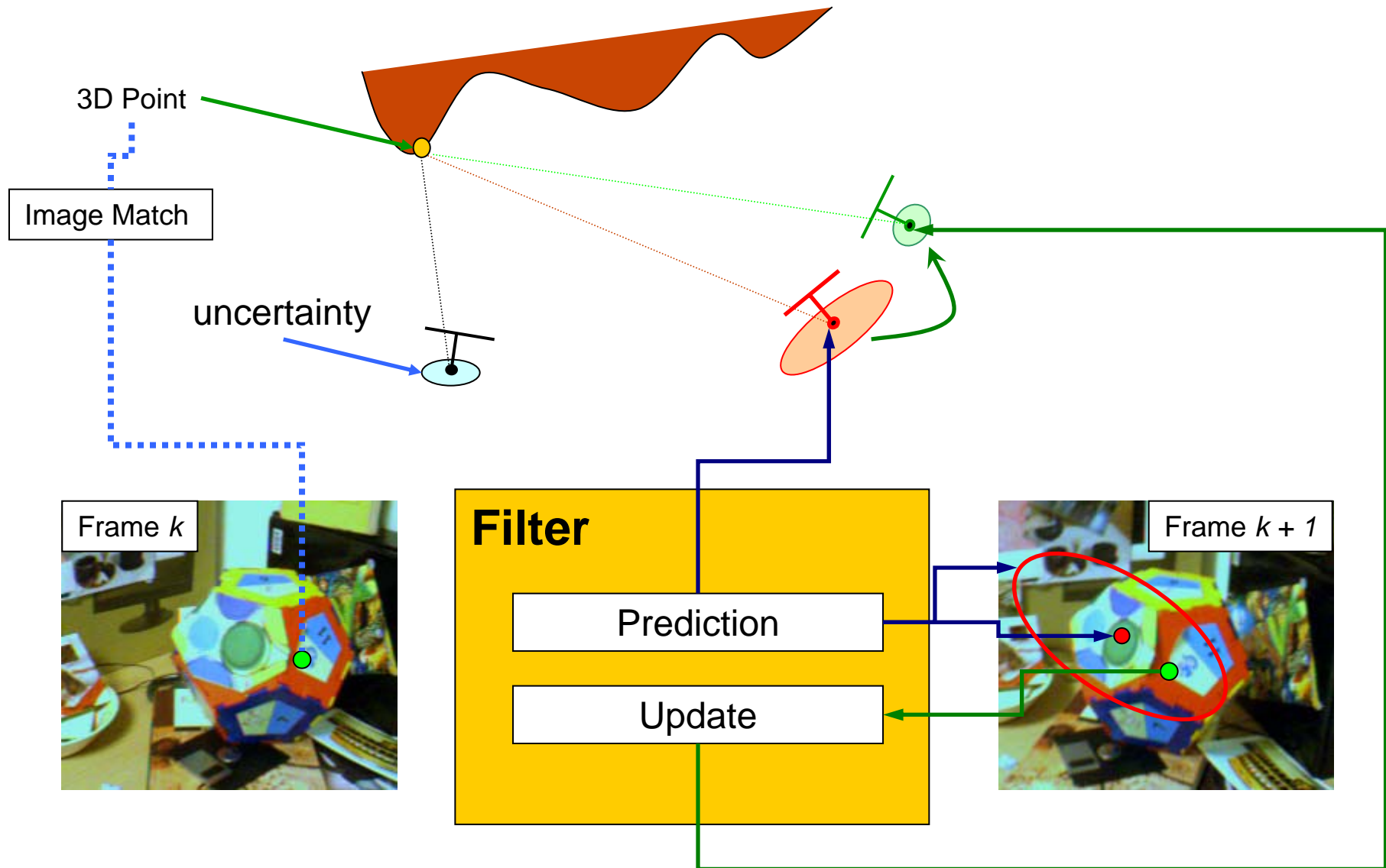
Correspondence Problem in SLAM

- Point identification (matching) is crucial for SLAM
- It needs to be done reliably and extremely fast
- Reliable matching is hard especially in real-time: ambiguity, occlusions, change in viewpoint
- Data is often noisy and prone to mismatches
- How can we deal with these?



Example
of ambiguous
features

SLAM Using Stochastic Filter



Stochastic Filters:

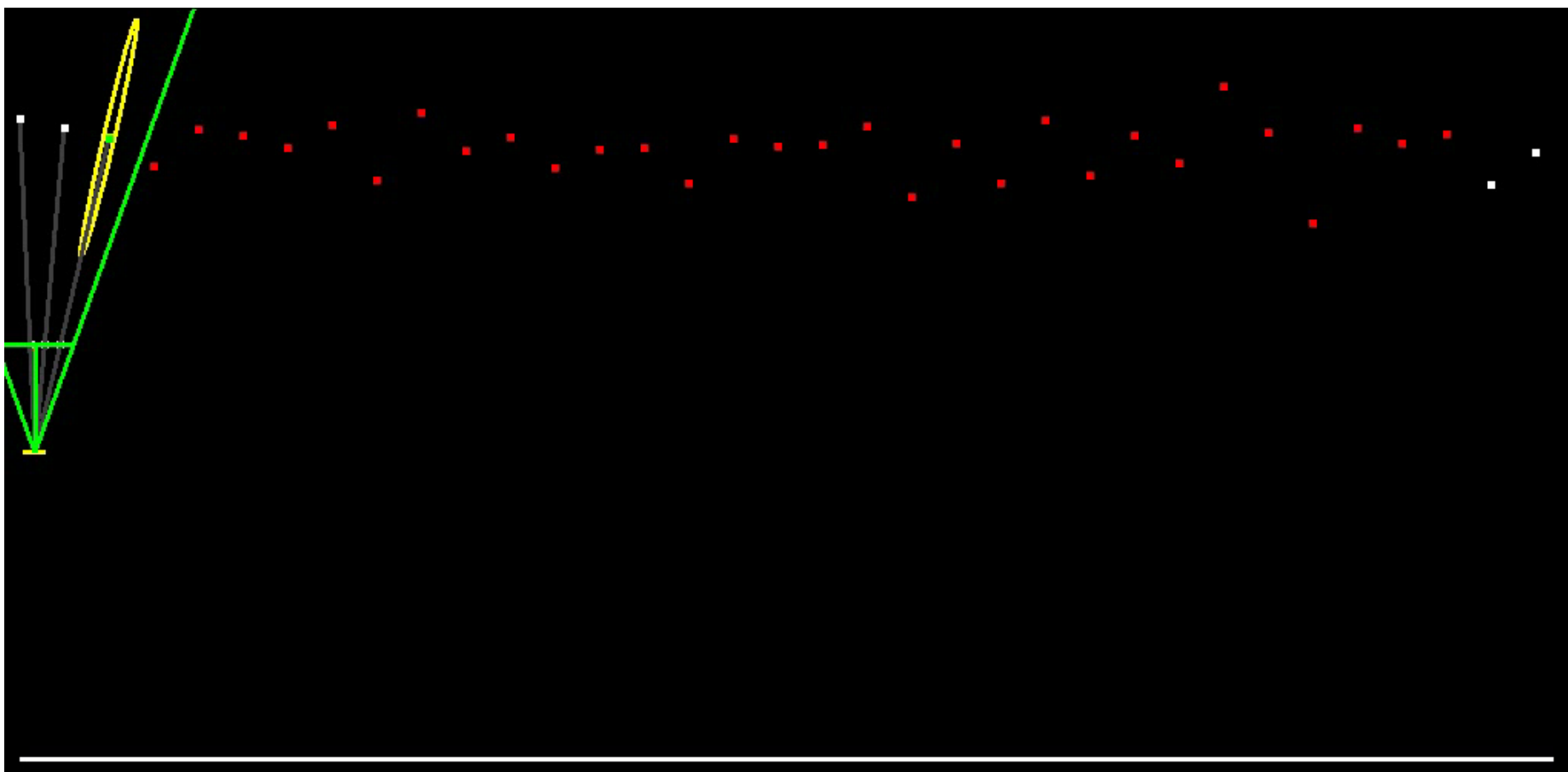
- Particle Filter
- Kalman Filter
 - Extended Kalman Filter – EKF (linearisation errors, but N^2)
 - Unscented Kalman Filter – UKF (easier to implement, but N^3) – *our choice currently*

Kalman Filter – state estimation algorithm
with prediction/correction structure

- State vector: $x_k = \{ \underbrace{t, q}_{\text{camera}}, \underbrace{p_1, \dots, p_N}_{\text{map}} \}$

- Covariance Matrix: P_{xx}
 - Characterises uncertainty of each component
 - Maintains stochastic links between components

Example: Importance of Covariance

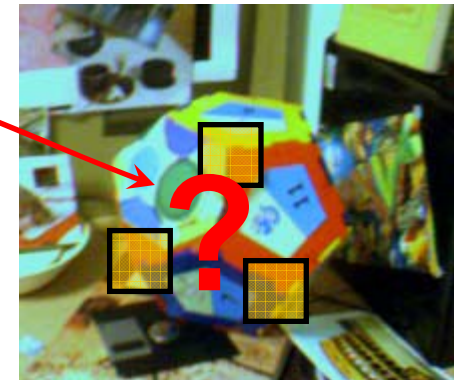


Solving Correspondence Problem

Image template matching



Extract image
Template
and try to
find a correspondence



- Conventionally template matching is performed using Normalised Cross-Correlation (NCC)
- NCC is simple and reasonably fast
- **But** NCC is NOT discriminative enough – **doesn't help to recover after erratic motions**

Solving Correspondence Problem

- *One solution* – use of multi-hypothesis stochastic filter (Particle Filter, Mixture of Gaussian)
- *Another way* – highly robust feature descriptors
- Popular descriptors (e.g. in SIFT) involve important, but **SLOW** detection stage
- We implemented new stochastic framework using descriptors based on spatial gradient histograms (similar to descriptors of SIFT)

Example: Importance of Features

SLAM using proposed multi-resolution feature descriptors

versus

SLAM using conventional template-based features and Normalised Cross-Correlation(NCC)



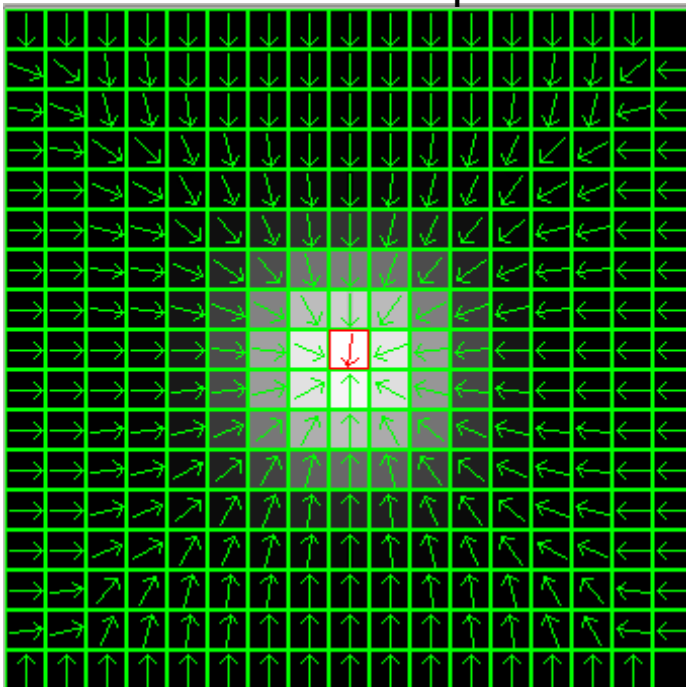
New feature descriptors



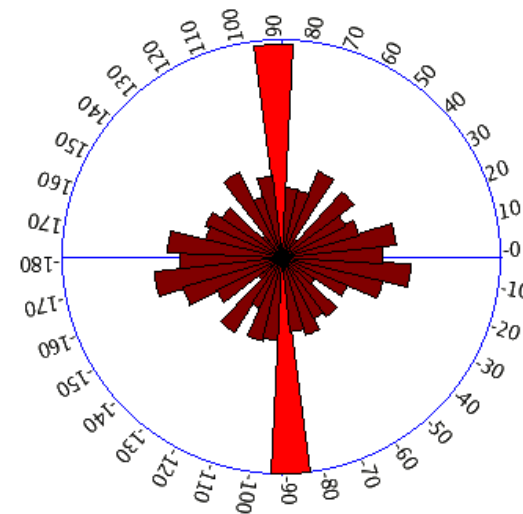
NCC

Spatial Gradients Descriptor

16x16 Gradient Field
around selected pixel

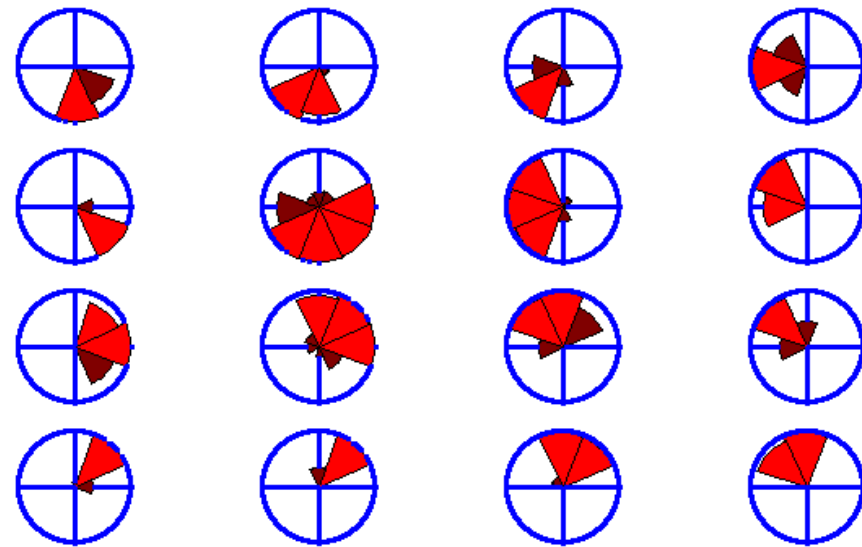


Orientation histogram over whole region
in order to detect dominant orientations

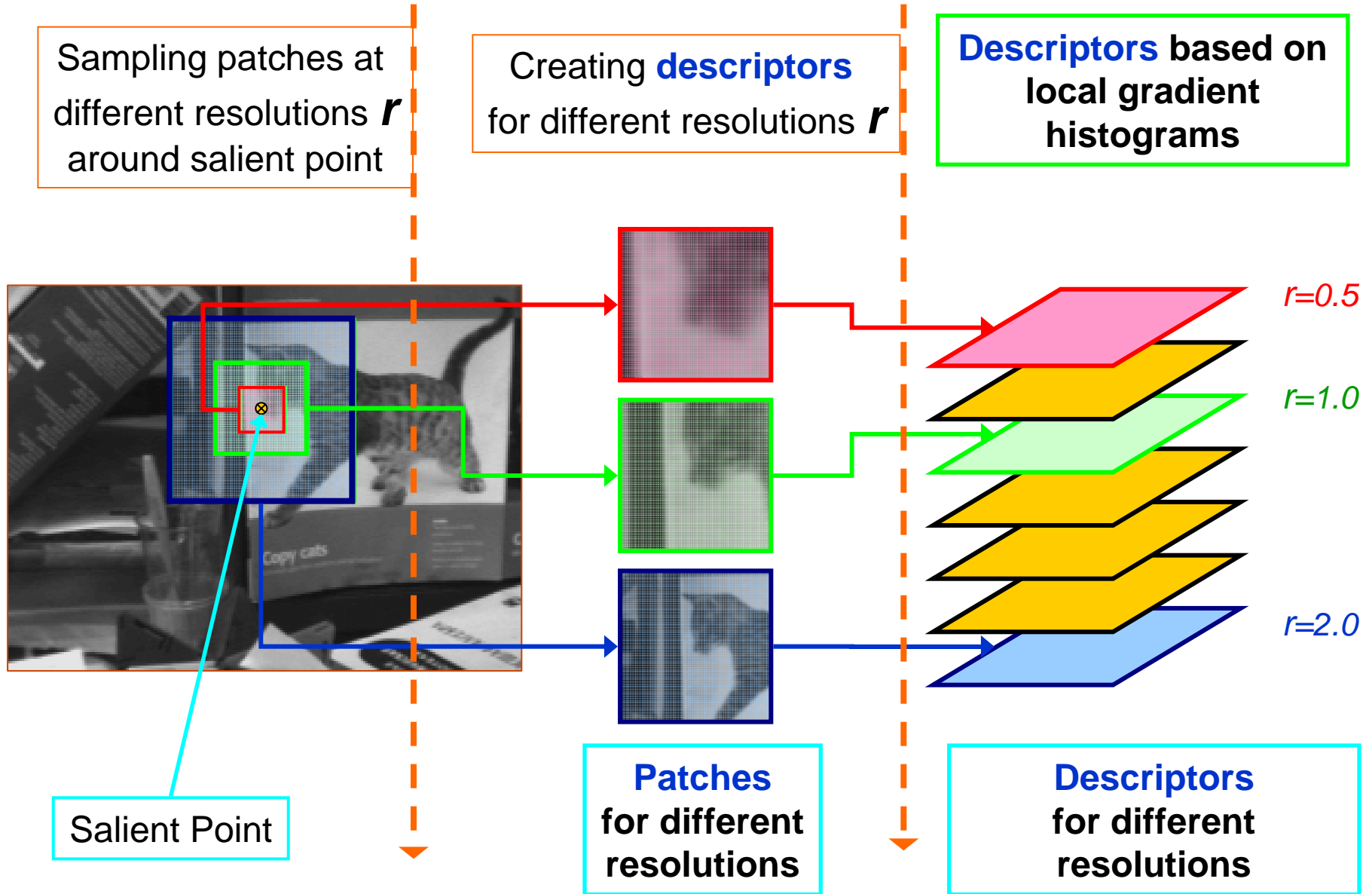


Orientation histograms (8 bins) over
4x4 pixel blocks, forming 4x4 matrix
of histograms.

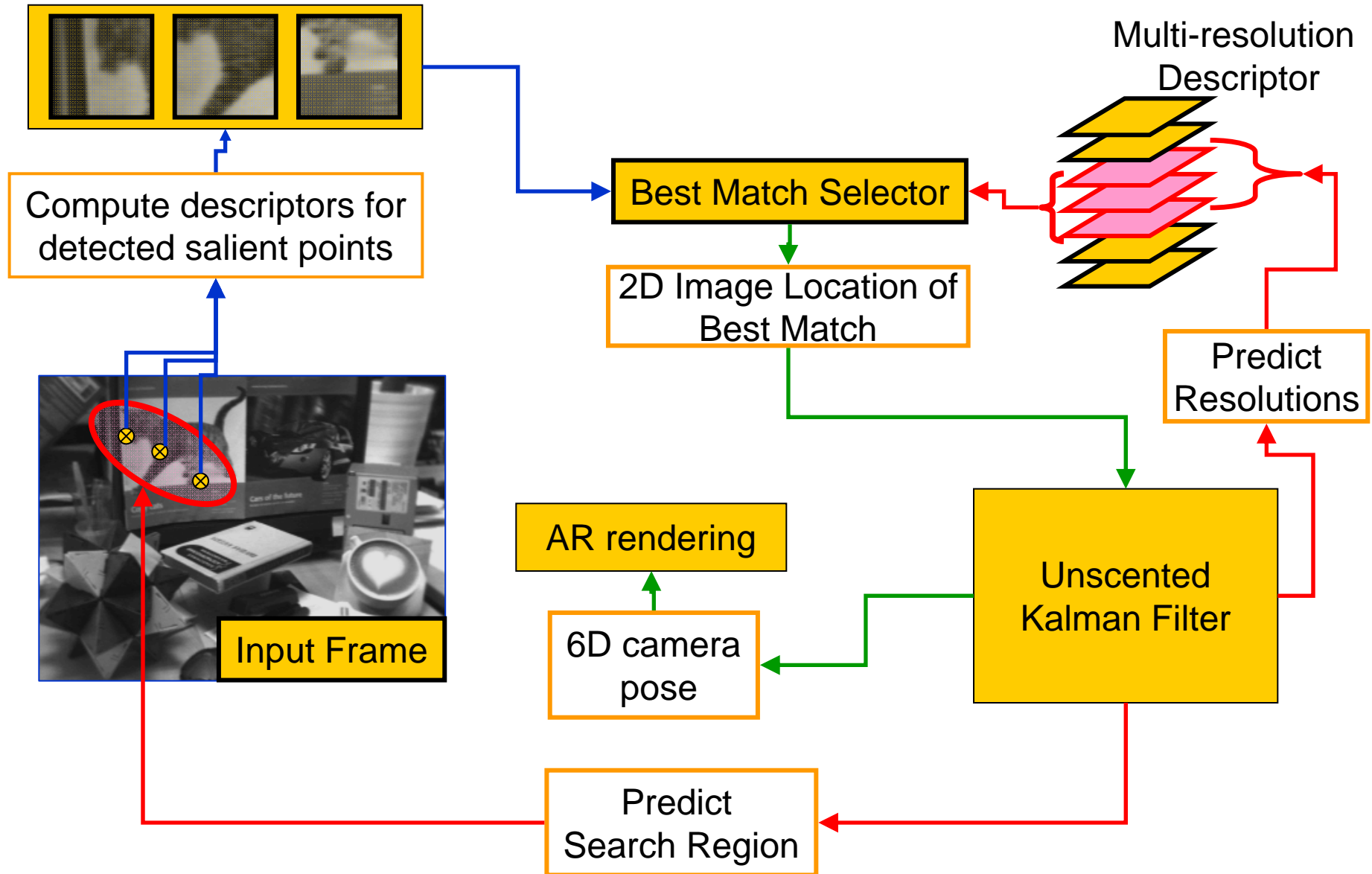
Values of histograms stacked into
128-sized vector, which is then
normalised.



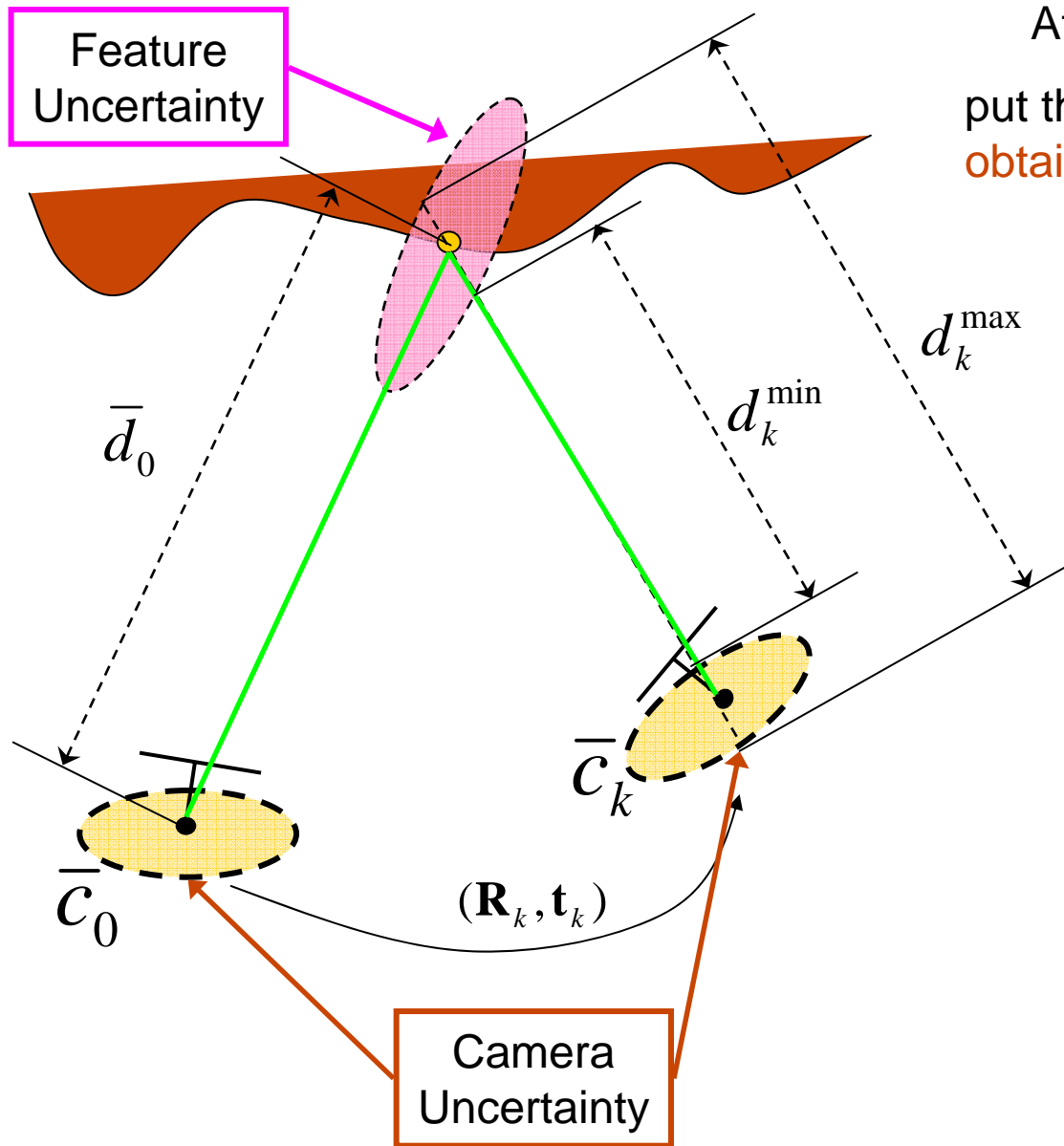
Multi-resolution Descriptor: Initialisation



System outline



Resolution Prediction



At each step k , depth to feature is put through **Unscented Transform** to obtain

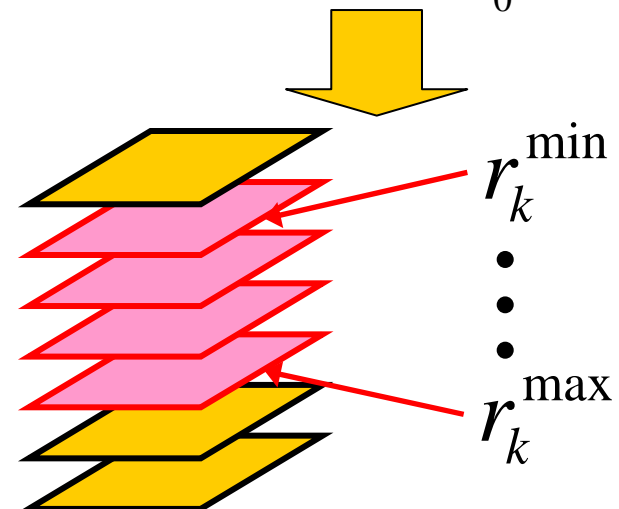
$$\rightarrow (\bar{d}_k, \sigma_k^d)$$

$$d_k^{\min} = \bar{d}_k - 3\sigma_k^d$$

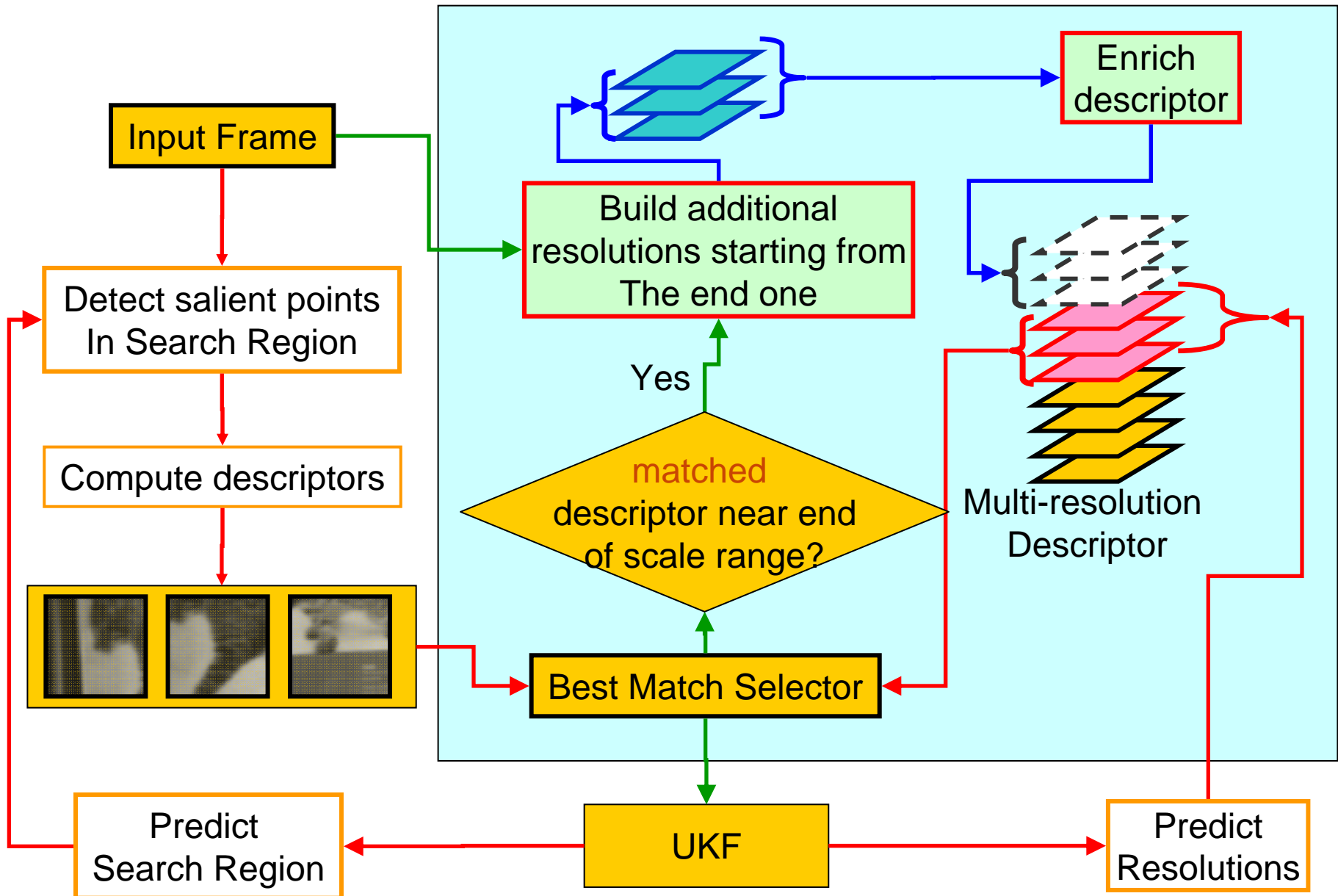
$$d_k^{\max} = \bar{d}_k + 3\sigma_k^d$$

Range of predicted resolution:

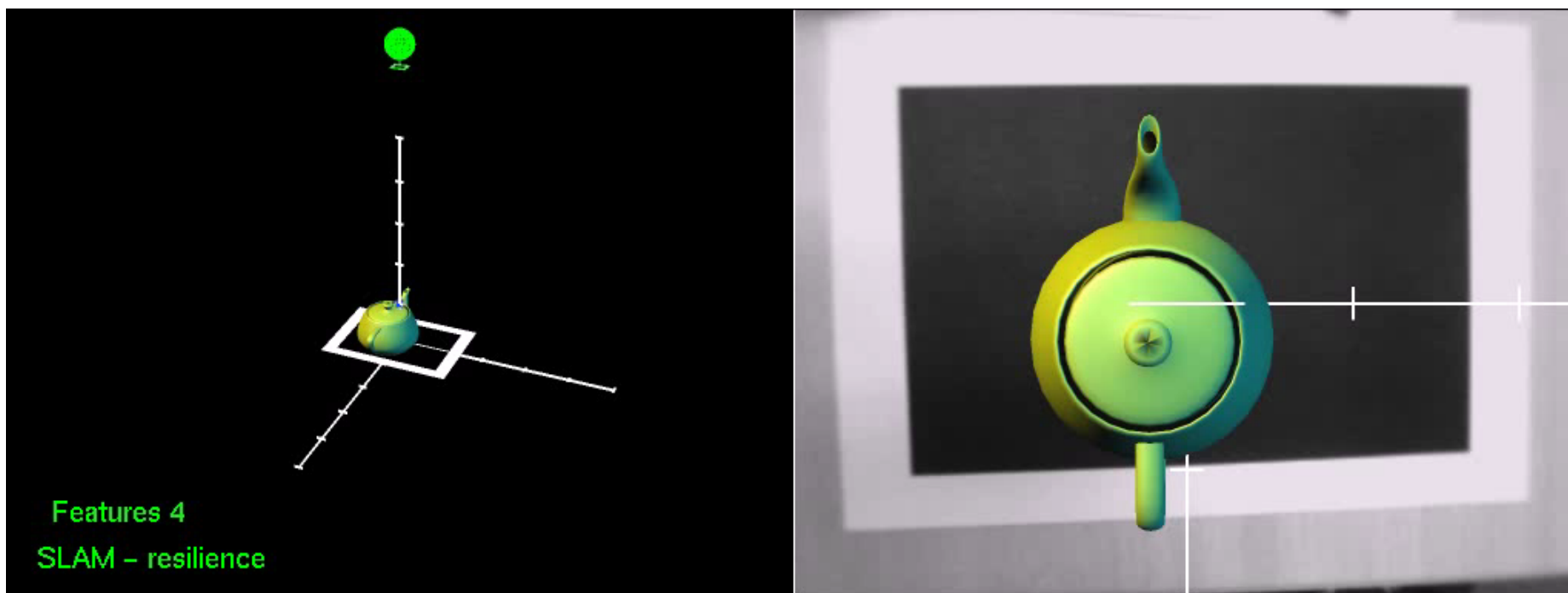
$$[r_k^{\min}, r_k^{\max}] = \left[\frac{d_k^{\min}}{\bar{d}_0}, \frac{d_k^{\max}}{\bar{d}_0} \right]$$



Building new scales



Example: Shake and Occlusion



Typical Timing

	10 features	20 features
Tracking	20 ms	30 ms
Adding Feature	45 ms	65 ms

- Improve efficiency of the system (active search, EKF)
- Deal with bigger maps
- Kidnap camera problem
- Combining sub-maps with unknown scale factor
- Take the system outdoors

- THE END