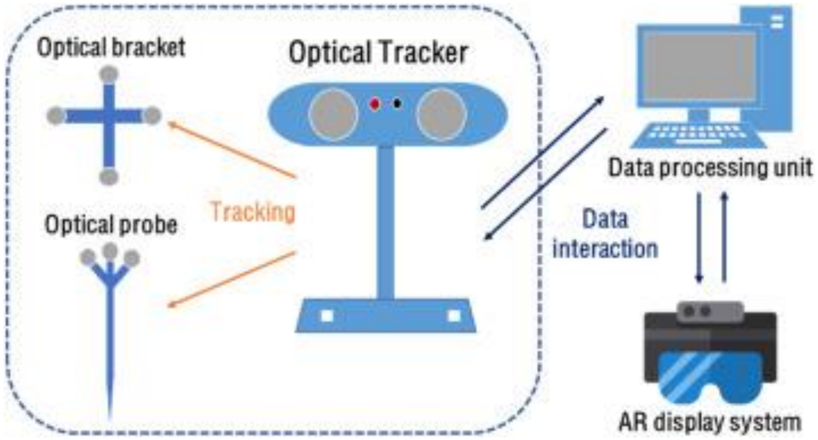
증강현실

(2023. 10. 16.)

이 종 원
(jwlee@sejong.ac.kr)



Optical Tracking

Why Optical Tracking for AR?

✓Many AR devices have cameras

- Mobile phone/tablet, video see-through display

✓Provide precise alignment between video and AR overlay

- Using features in a video to generate pixel-perfect alignment
- The real world has many visual features that can be tracked from

✓Computer vision is a well-established discipline

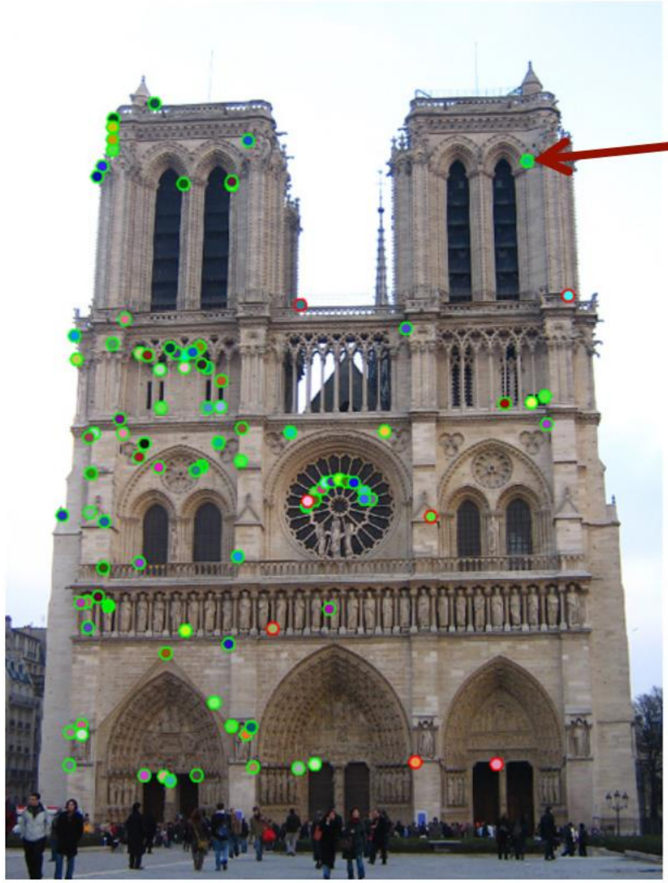
- Over 40 years of research
- Old non-real-time algorithms can be run in real-time on today's devices

Visual Features

- ✓Keypoints and descriptors together define common visual features
- ✓A keypoint is a (locally) distinct location in an image
- ✓The feature descriptor summarizes the local structure around the keypoint

Slides created by Cyrill Stachniss as part of the Photogrammetry courses taught in 2014 and 2019.

Keypoint and Descriptor



keypoint

descriptor at the keypoint

$$f = \begin{bmatrix} 0.02 \\ 0.04 \\ 0.1 \\ 0.03 \\ 0 \\ \dots \end{bmatrix}$$

Slides created by Cyrill Stachniss as part of the Photogrammetry courses taught in 2014 and 2019.

Keypoints

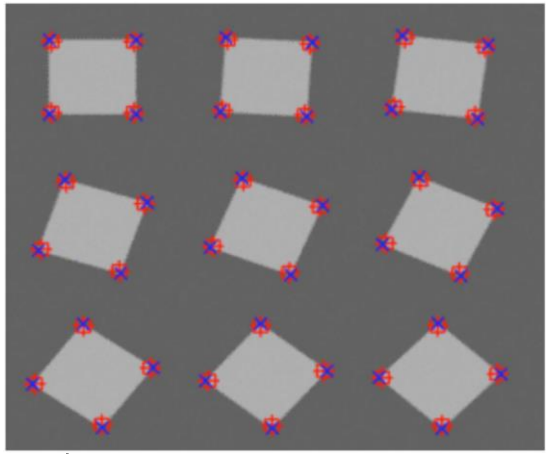
✓Finding locally distinct points

✓Harris: Corners

✓Shi-Tomasi: Corners

✓Difference of Gaussians:

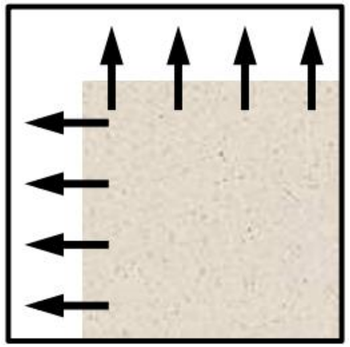
- Corners, edges, and blobs



Slides created by Cyrill Stachniss as part of the Photogrammetry courses taught in 2014 and 2019.

Corners and Edges

✓Corners are invariant to translation, rotation, and illumination



✓Corner = two edges (in roughly orthogonal directions)

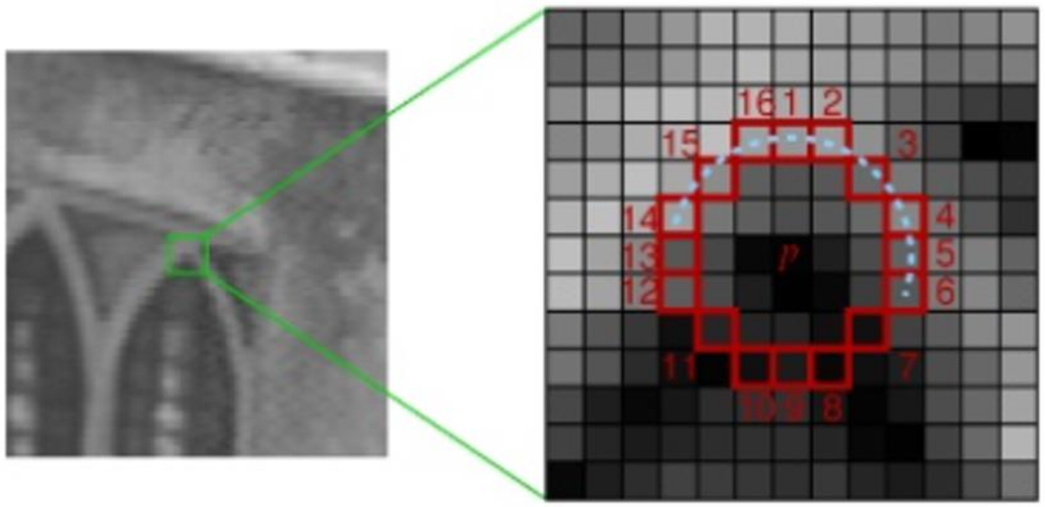
- Search for intensity changes in two directions

✓Edge = a sudden brightness change

Slides created by Cyrill Stachniss as part of the Photogrammetry courses taught in 2014 and 2019.

FAST Corner Keypoint Detection

✓FAST (Features from Accelerated Segment Test)



FAST Corner Detection

https://youtu.be/vEkHoYpMD3Y?si=tB1SHS5Los5OW4NU (2018, 1:31)

FAST Corner Detection

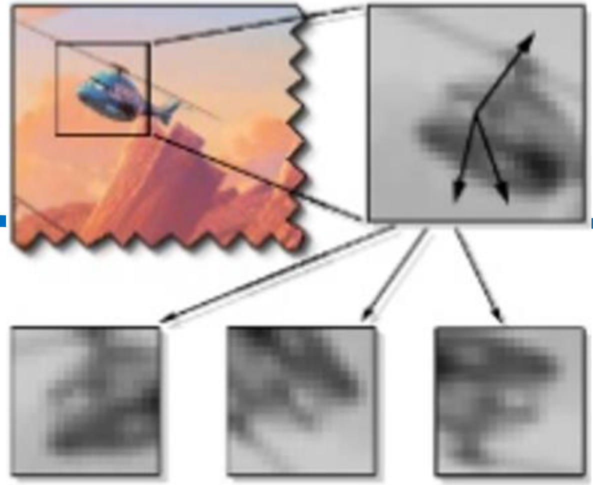
https://youtu.be/pJ2xSrlXy_s (2009, 0:31)

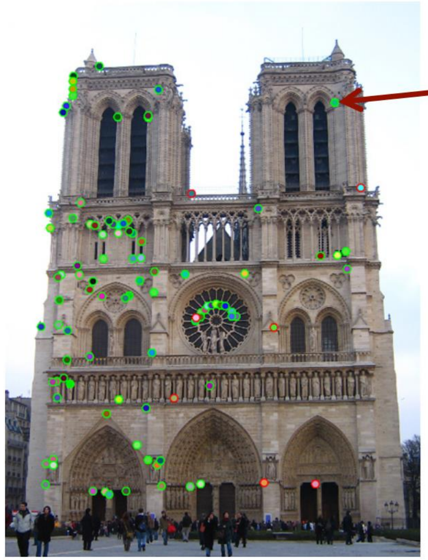
Descriptors

✓Keypoint features only contain position information

✓Descriptors describe the keypoint features

- Vector describing the information of the pixels around the keypoints (e.g., orientation, size)





$$f = \begin{bmatrix} 0.02 \\ 0.04 \\ 0.1 \\ 0.03 \\ 0 \\ \ldots \end{bmatrix}$$

Wagner D., Reitmayr G., Mulloni A., Drummond T., Schmalstieg D., Real-time detection and tracking for augmented reality on mobile phones. IEEE Transactions on Visualization and Computer Graphics, May/June 2010

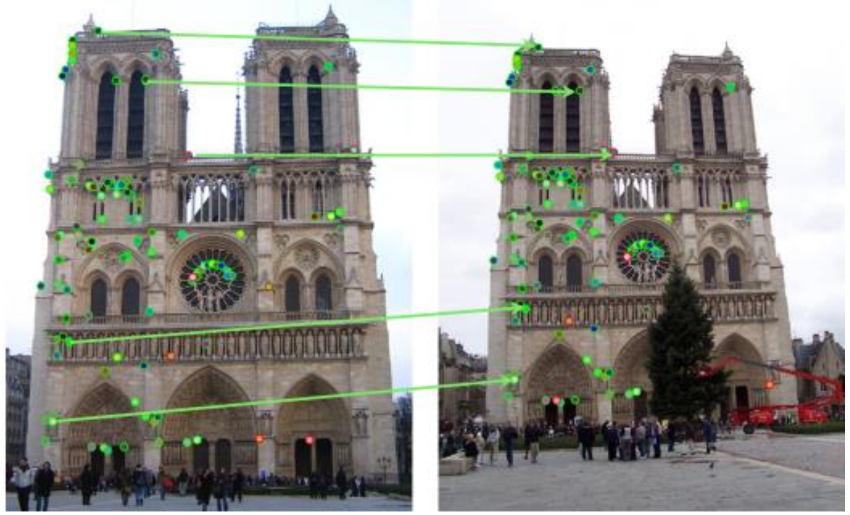
Descriptors

✓Describing a keypoint

✓SIFT: Scale Invariant Feature Transform

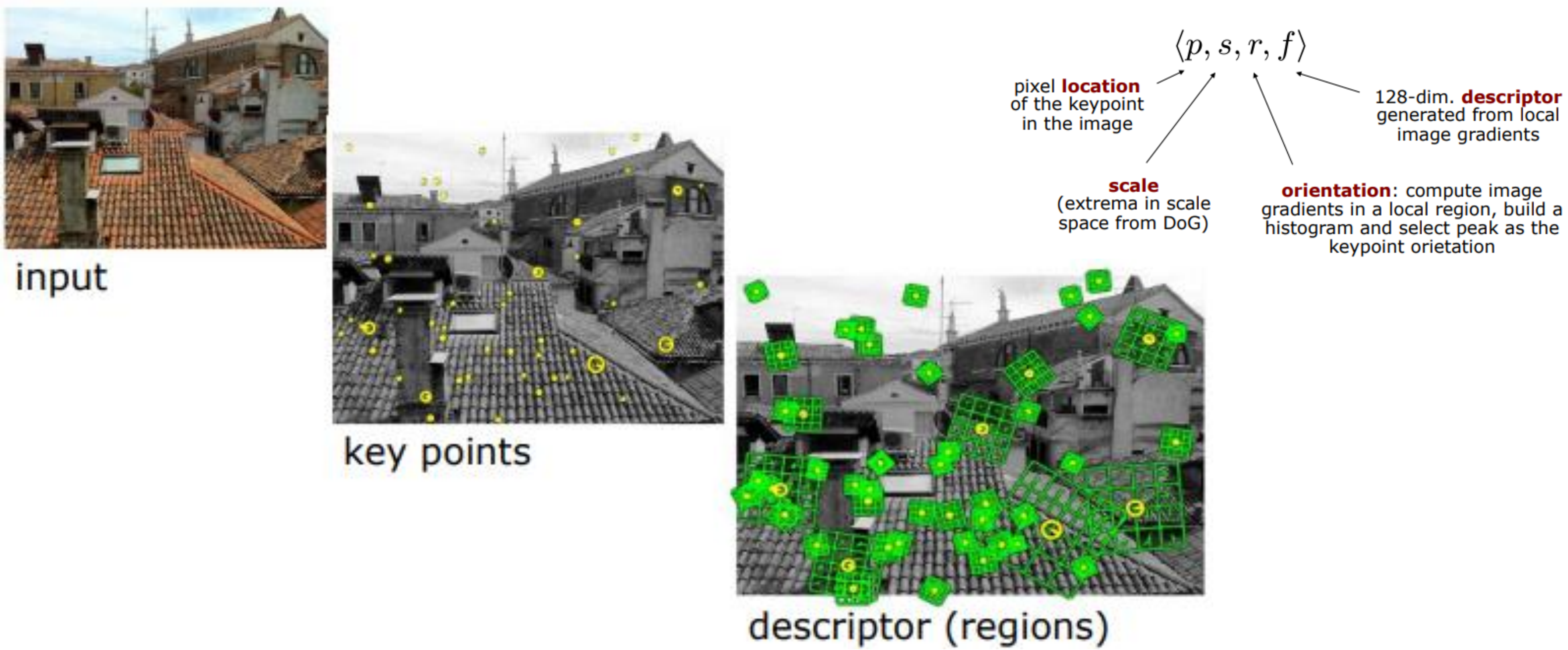
✓BRIEF: Binary Robust Independent Elementary Features

✓ORB: Oriented FAST Rotated BRIEF



Slides created by Cyrill Stachniss as part of the Photogrammetry courses taught in 2014 and 2019.

SIFT Descriptor



Slides created by Cyrill Stachniss as part of the Photogrammetry courses taught in 2014 and 2019.

Common AR Optical Tracking Types

✓Marker tracking

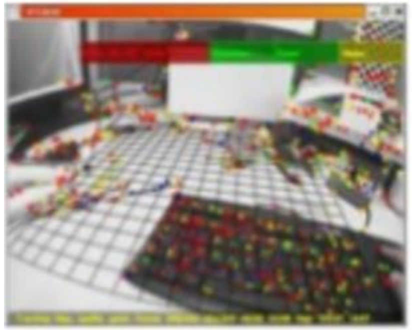
- Tracking known artificial markers/images
- E.g., ARToolKit square markers

✓Markerless tracking

- Tracking from known features in real world
- E.g., Vuforia image tracking

✓Unprepared tracking

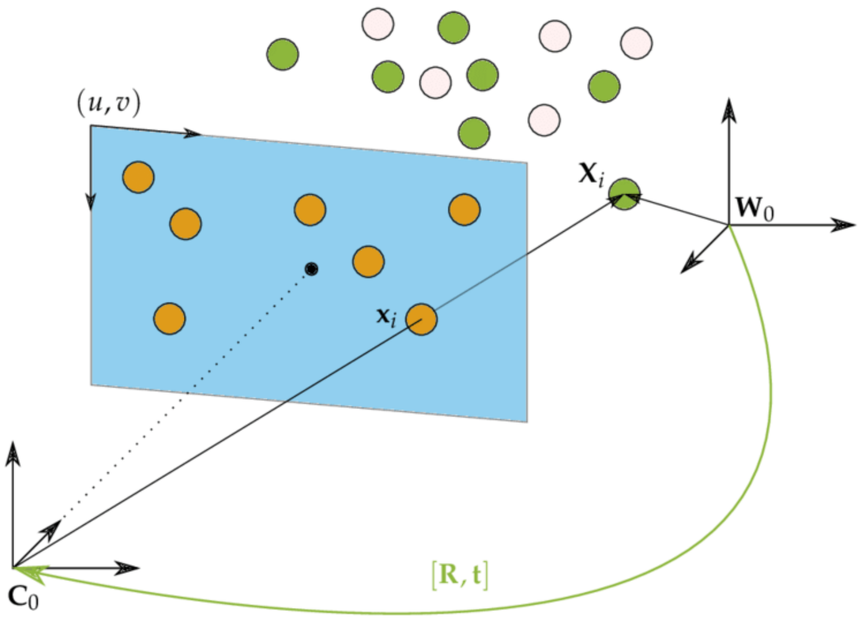
- Tracking in unknown environment
- E.g., SLAM tracking



Estimating an Image Pose

✓Given a set of 3D points of a scene and their corresponding 2D points in an image

✓The image pose is computed from corresponding 2D points (orange) and 3D points (green)



Markus Műller, “Camera Re-Localization with Data Augmentation by Image Rendering and Image-to-Image Translation,” PhD Thesis, 2020

Marker Tracking

✓Available for more than 20 years

✓Several open-source solutions

- ARToolKit, ARTag, ATK+, …

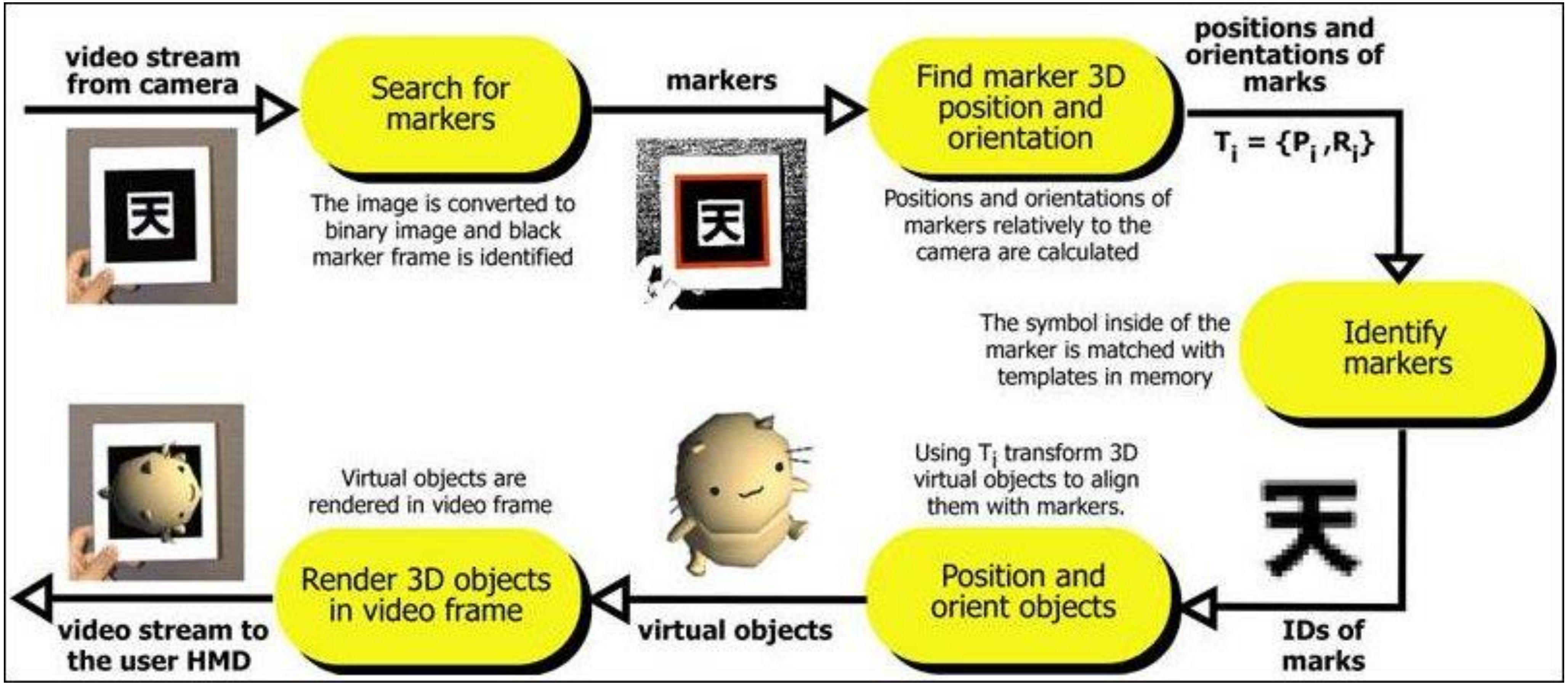
✓Fairly simple to implement

- Standard computer vision methods

✓A rectangle provides 4 corner points

- Enough for pose estimation

Marker Based Tracking: ARToolKit



Tracking Challenges in ARToolKit

Occlusion
(image by M. Fiala)

Unfocused camera,
motion blur

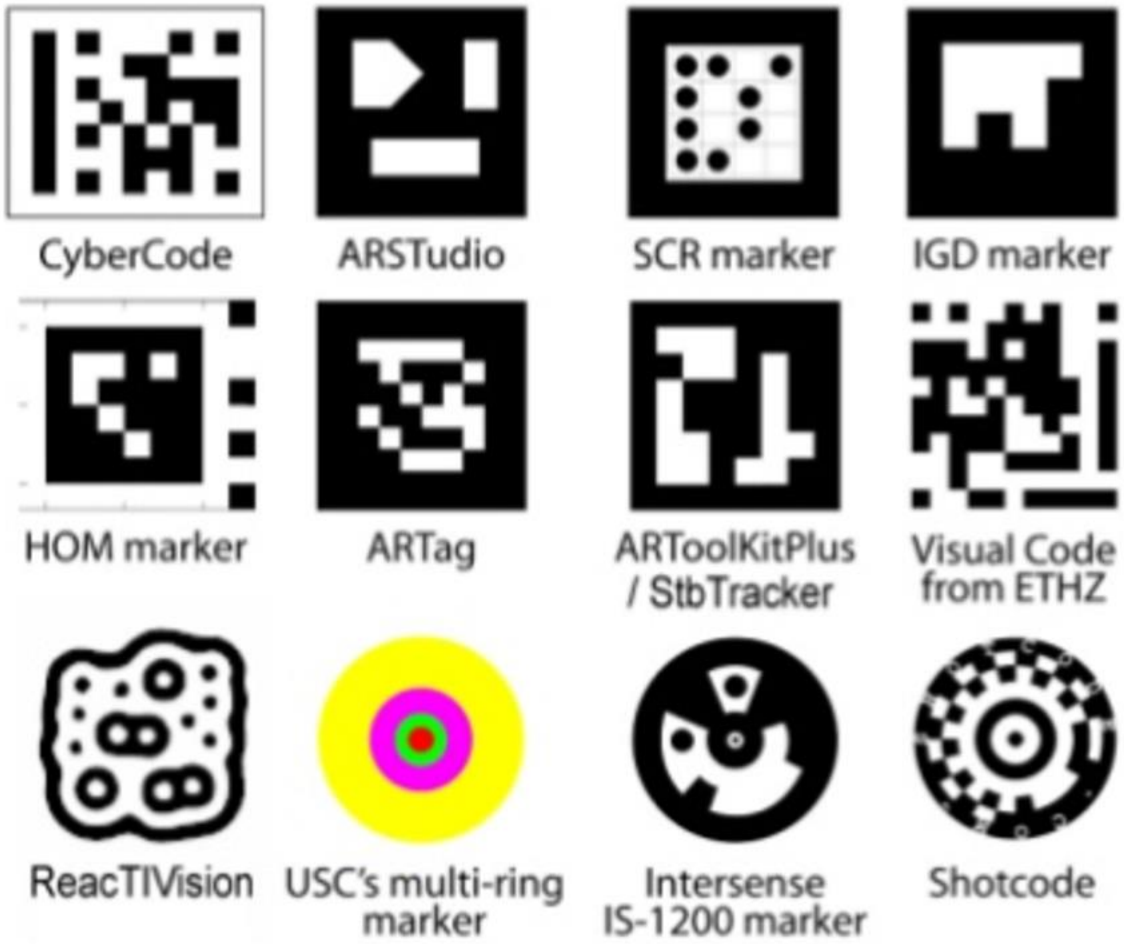
Dark/unevenly lit
scene, vignetting

Jittering
(Photoshop illustration)

False positives and inter-marker confusion
(image by M. Fiala)

Image noise
(e.g. poor lens, block
coding /

Other Marker Tracking Libraries



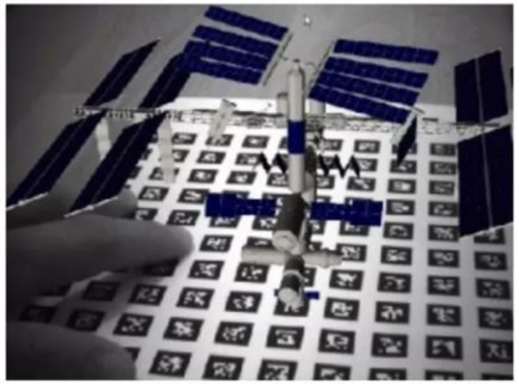
Marker Target Identification

✓More targets or features → more easily confused

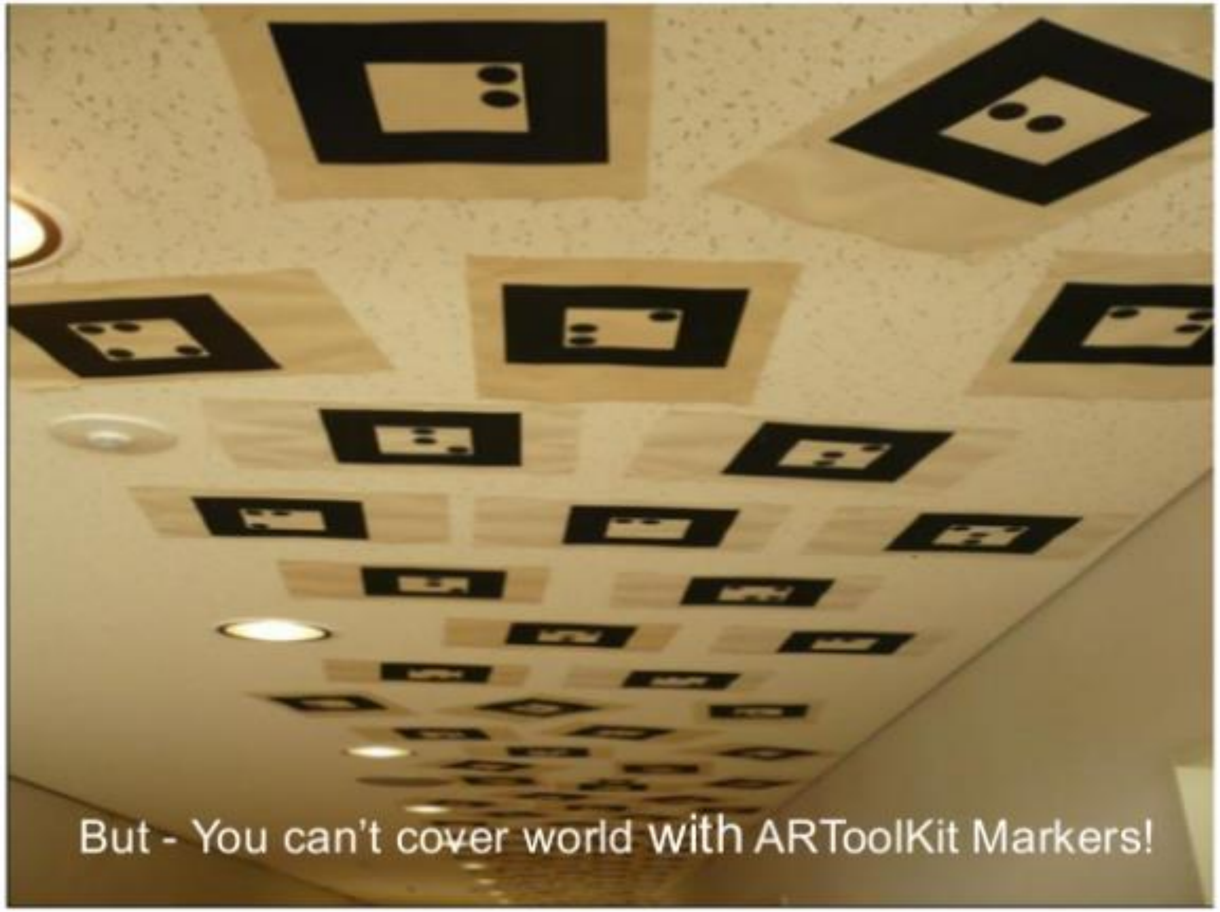
✓Must be as unique as possible

✓Square markers

- 2D barcodes with error correction
- E.g., 6 x 6 = 36 bits (2 orientations, 6-12 payload, rest for error correction)

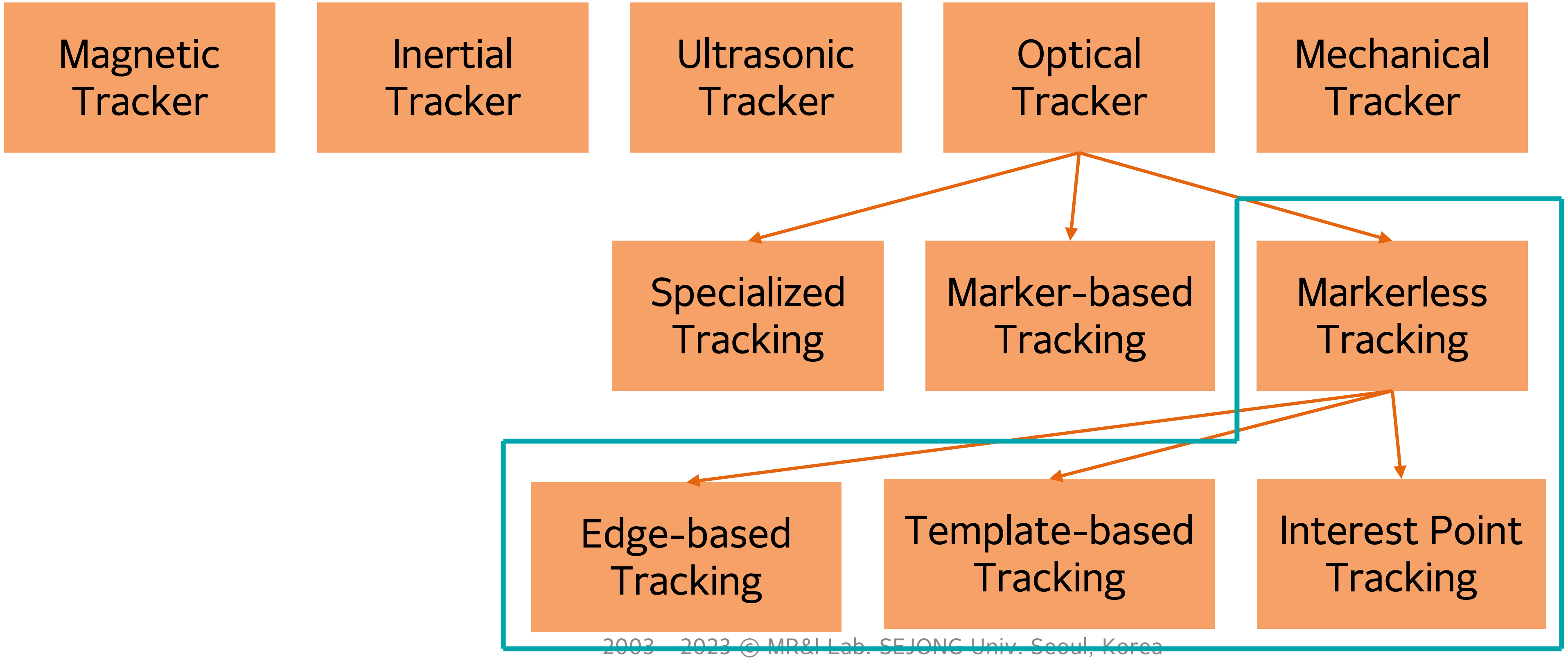


Can’t Cover World



Markerless Tracking

✓No more markers! → Markerless tracking



Natural Feature Tracking

✓Use natural cues of real elements

- Edges
- Surface textures
- Interest points

✓Model or model-free

✓No visual pollution

Texture Tracking

Natural Features

✓ Detect salient interest points in an image

- Must be easily found
- Location in the image should remain stable when the viewpoint changes
- Require textured surfaces
- Alternative: can use edge features (less discriminative)

✓ Match interest points to tracking model database

- Database filled with results of 3D reconstruction
- Matching entire (sub-)images is too costly
- Typically, interest points are compiled into “descriptors”

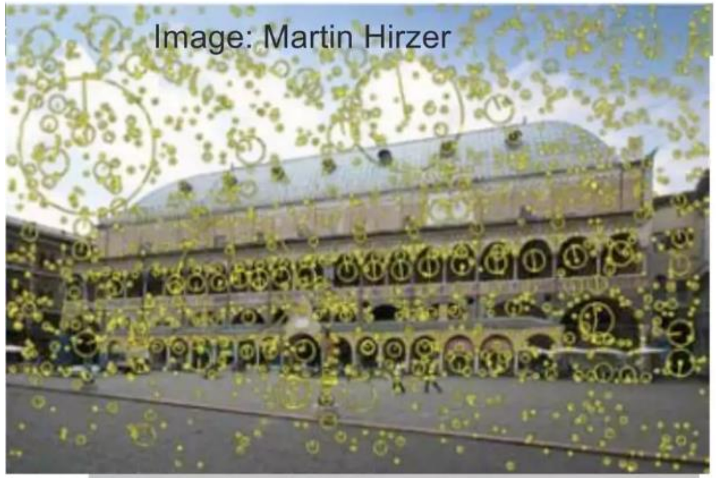


Image: Martin Hirzer

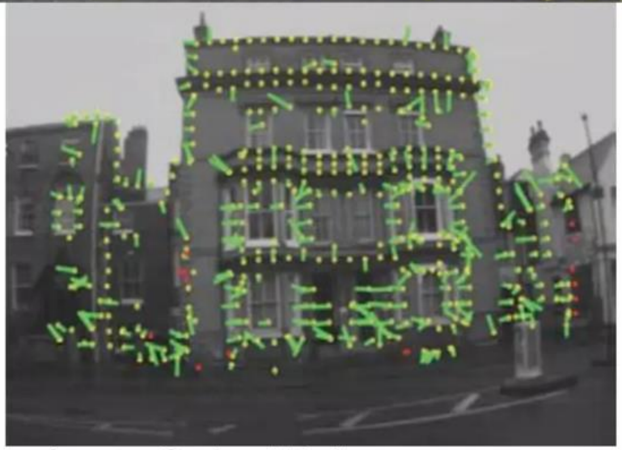


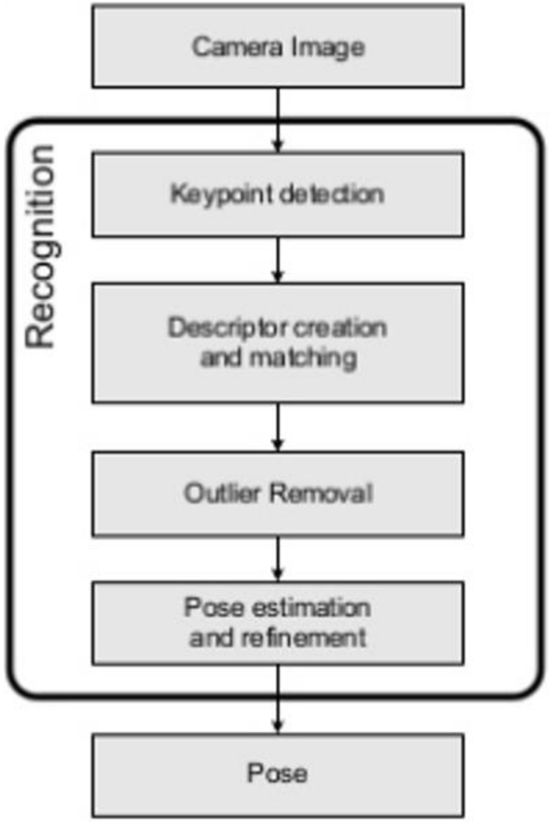
Image: Gerhard Reitmayr

Tracking by Keypoint Detection

✓This is what most trackers do …

✓Targets are detected every frame

✓Popular because tracking and detection are solved simultaneously

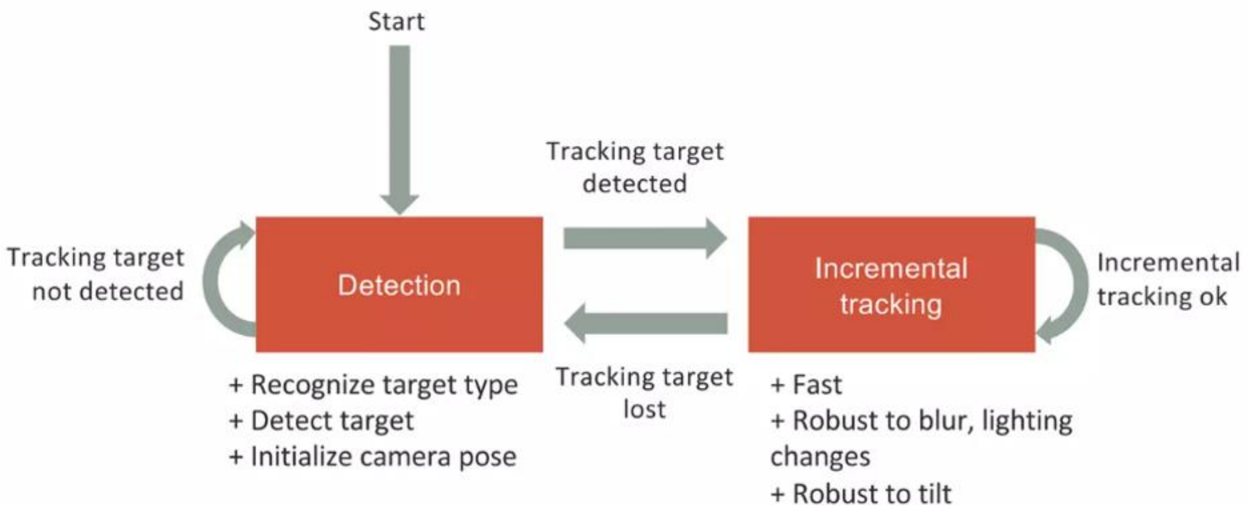


Detection and Tracking

✓Tracking and detection are complementary approaches

✓After successful detection, the target is tracked incrementally

✓If the target is lost, the detection is activated again



Vuforia Texture Tracking

https://youtu.be/1Qf5Qew5zSU

Marker vs. Natural Feature Tracking

✓Marker tracking

- Markers can be an eye-catcher
- Tracking is less demanding
- The environment must be instrumented
- Markers usually work only when fully in view

✓Natural feature tracking

- Natural feature targets might catch the attention less
- Natural feature targets are potentially everywhere
- Natural feature targets also work if partially in view

Visual Tracking Approaches

✓ Marker-based tracking with artificial features

- Make a model before tracking

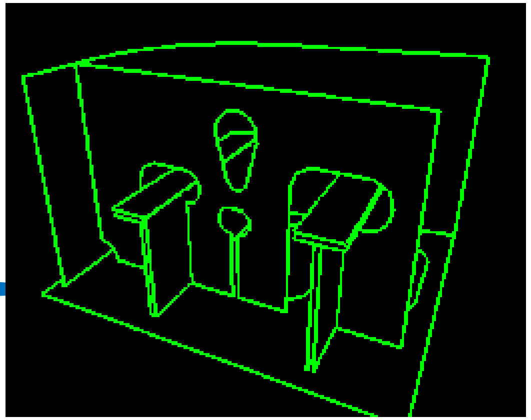
✓ Model-based tracking with natural features

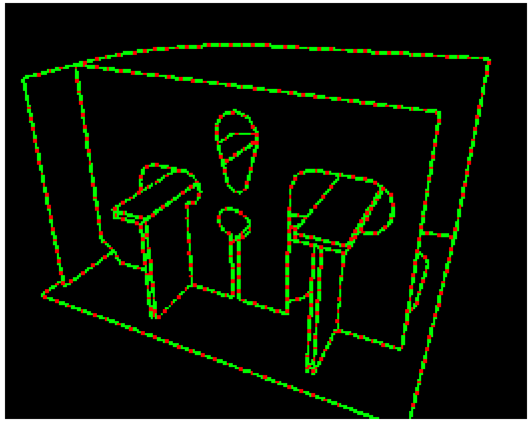
- Acquire a model before tracking

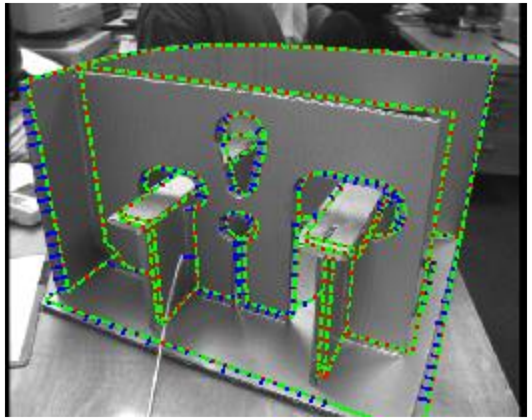
✓Simultaneous localization and mapping

- Build a model while tracking it

Edge/Model Based Tracking







✓Example: RAPiD [Drummond et al. 02]

- Initialization, control points, pose prediction (Global Method)

① Based on a CAD model of the structure

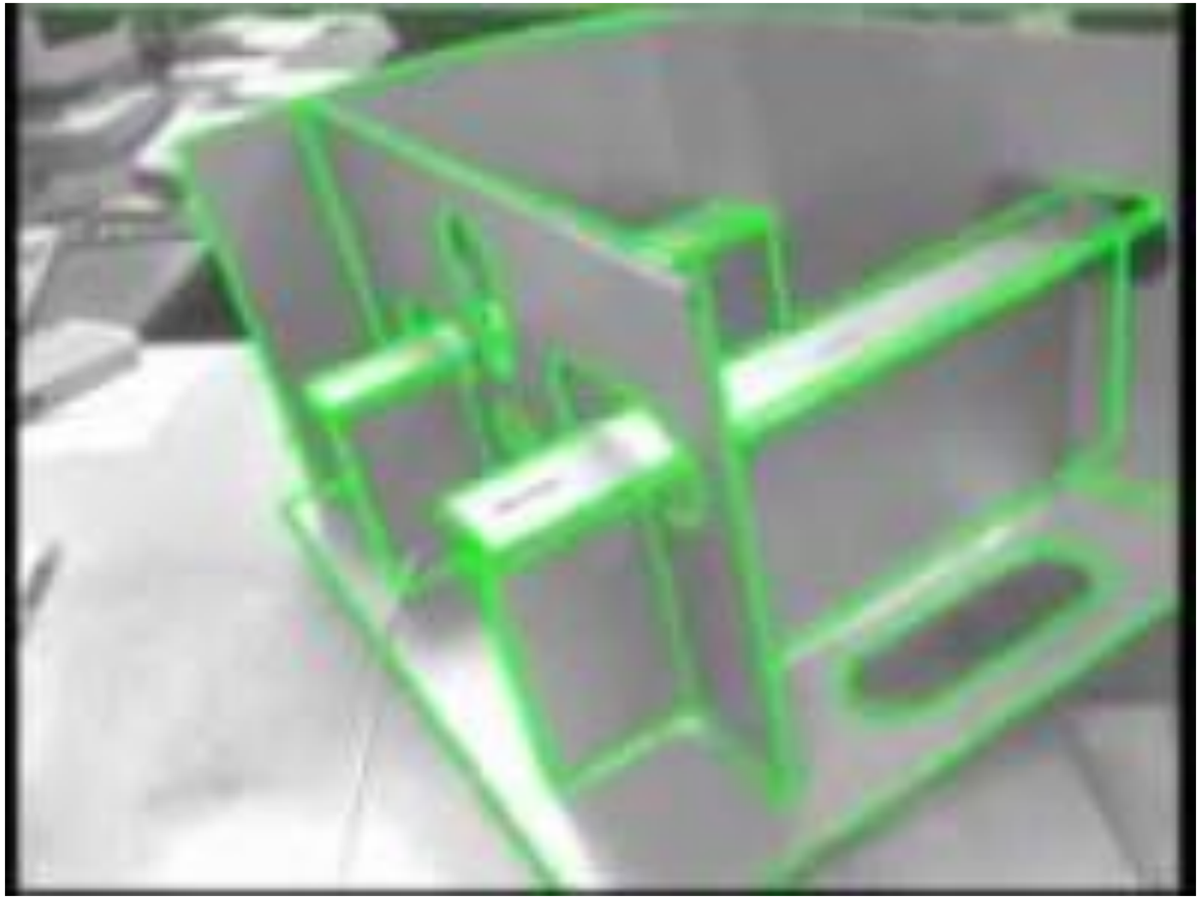
② The CAD model is rendered from a predicted viewpoint

③ Identify the edges of the model that are expected to be visible

④ The search is conducted from each sample point for an intensity discontinuity in the video image

⑤ A change in the pose is computed which minimizes these errors

Edge/Model Based Tracking



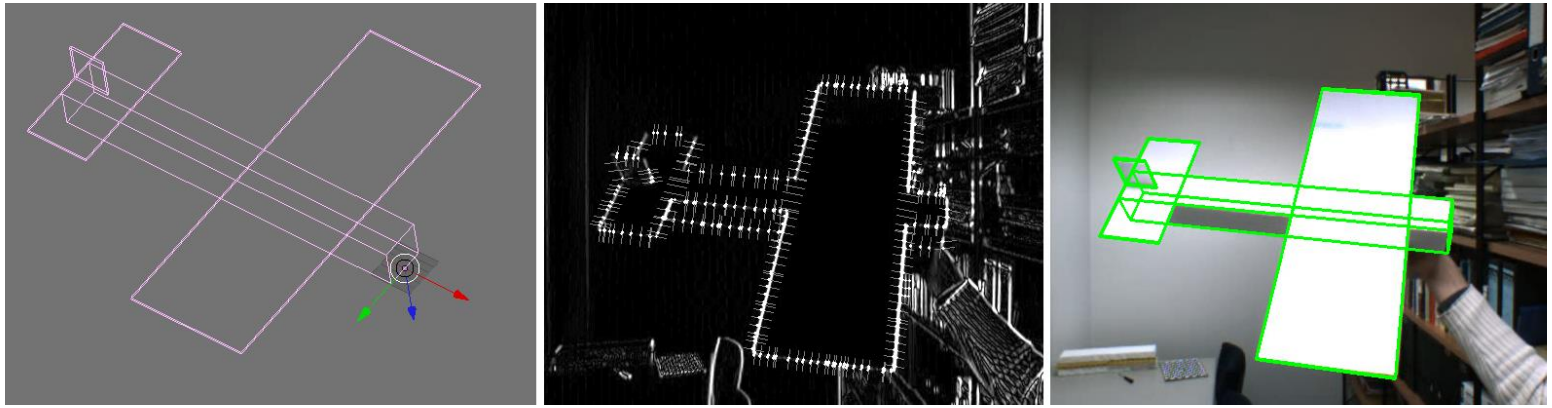
https://youtu.be/Z1J0VLizVDs

Model Based Tracking

✓Tracking from 3D object shape

✓Example: OpenTL

- General purpose library for model-based visual tracking



OpenTL Model Tracking

https://youtu.be/laiykNbPkgg

OpenTL Face Tracking

https://youtu.be/WIoGdhkfNVE

Vuforia Model Tracker

✓Uses pre-captured 3D model for tracking

✓On-screen guide to line up model

https://library.vuforia.com/model-targets/model-target-guide-view

How to Create Model Target

https://youtu.be/jbaUDMvv2Zw?si=W5-eDWe73upB2eFo (2020, 5:51)

Tracking from an Unknown Environment

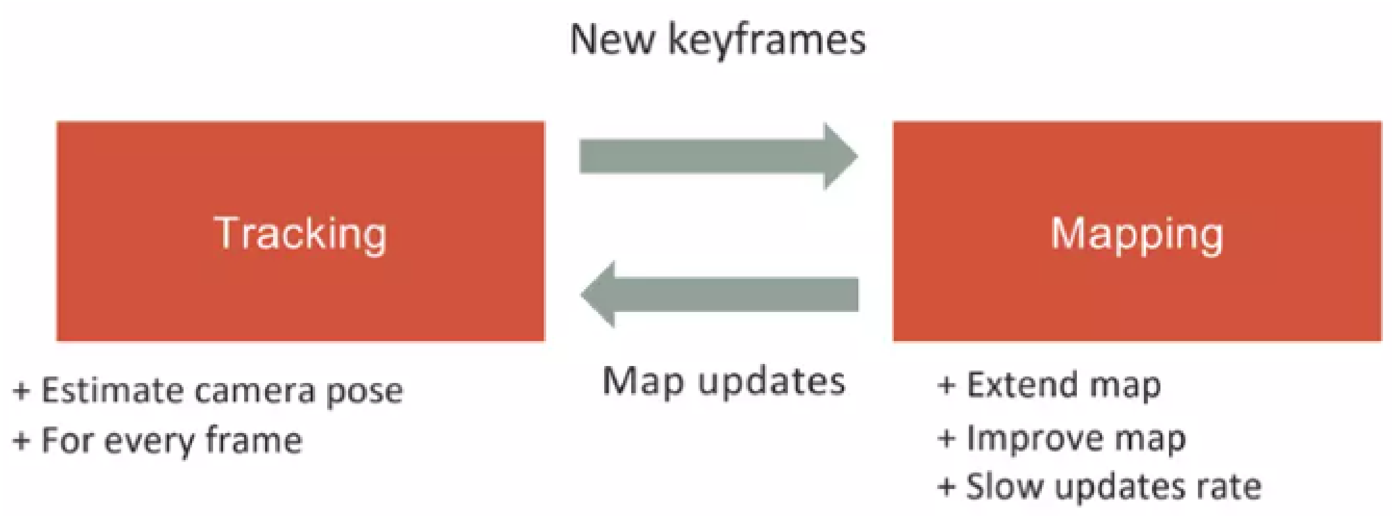
✓What to do when you don't know any features?

- Very important problem in mobile robotics – Where am I?

✓SLAM

- Simultaneously Localize And Map the environment
- Goal: Recover both camera pose and map structure while initially knowing neither
- Mapping: Build a map of the environment which the robot is in
- Localization: Navigate this environment using the map while keeping track of the robot's relative position and orientation

Parallel Tracking and Mapping



✓Parallel tracking and mapping uses two concurrent threads, one for tracking and one for mapping, which run at different speeds

Visual SLAM

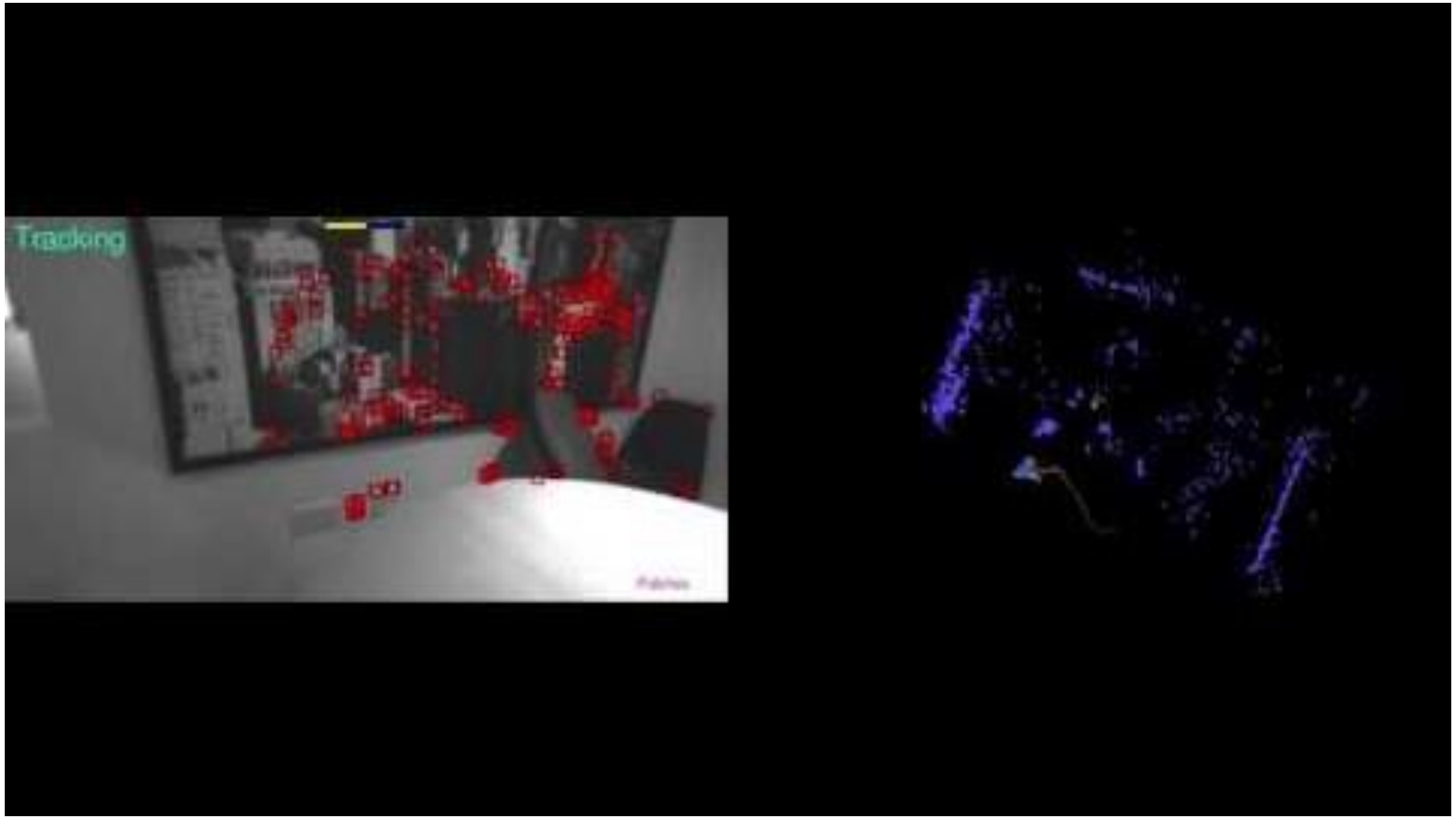
✓Early SLAM systems (1986~)

- Computer visions and sensors (e.g., IMU, laser, etc.)
- One of the most important algorithms in robotics

✓Visual SLAM

- Using cameras only, such as stereo view
- MonoSLAM (single camera) developed in 2007 (Davidson)

Kudan MonoSLAM



https://youtu.be/g2SFJGDz9cQ (0:58)

How SLAM Works

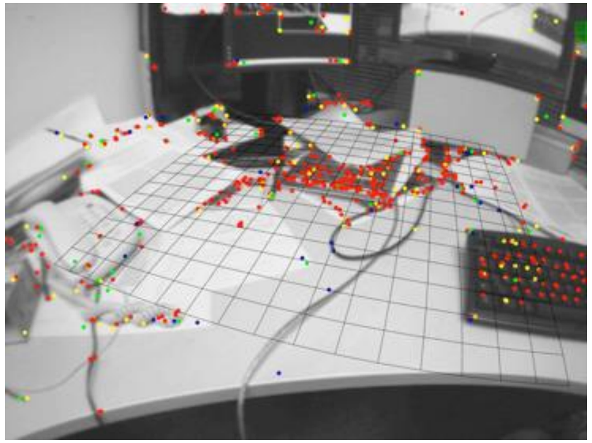
✓Three main steps

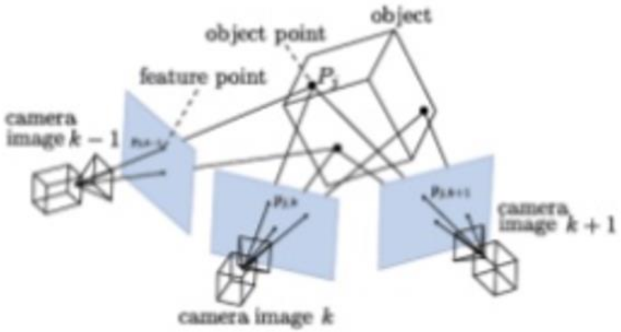
① Tracking a set of points through successive camera frames

② Using these tracks to triangulate their 3D position

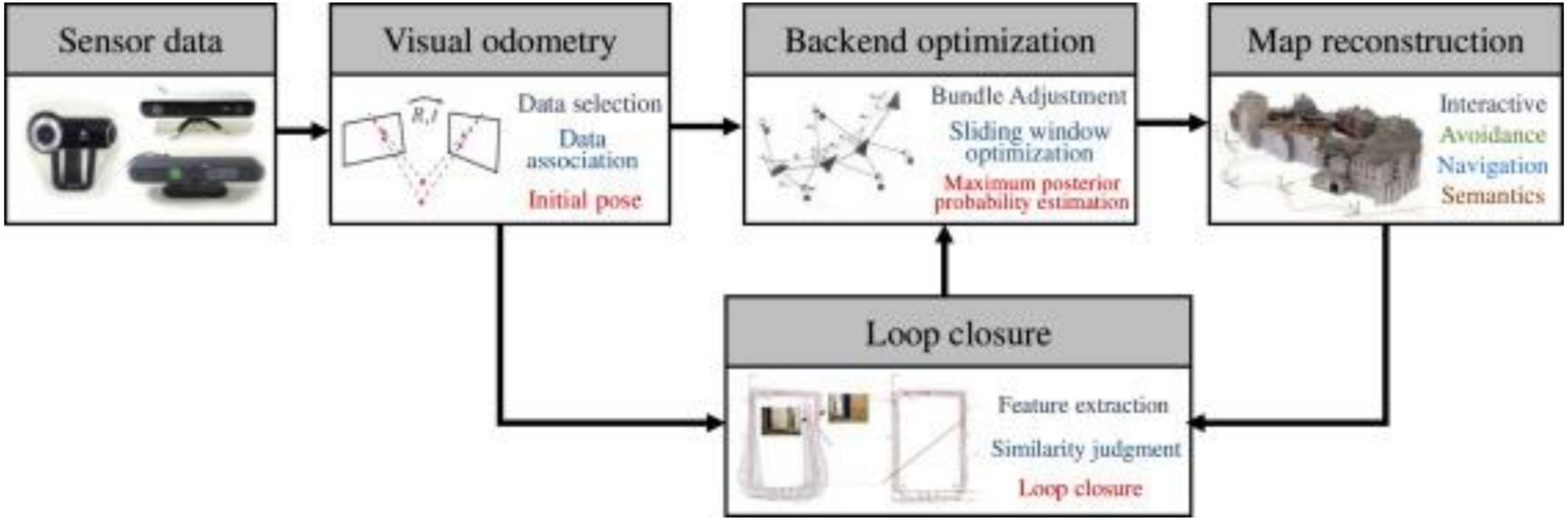
③ Simultaneously use the estimated point locations to calculate the camera pose which could have observed them

✓By observing enough points can solve for both structure and motion (camera path and scene structure)





Framework of Traditional VSLAM



Evolution of SLAM Systems

✓ MonoSLAM (Davidson, 2007)

- Real-time SLAM from a single camera

✓ PTAM (Klein, 2007)

- First SLAM implementation on mobile phone

✓ FAB-MAP (Fast Appearance Based Mapping, Cummins, 2008)

- Probabilistic localization and mapping

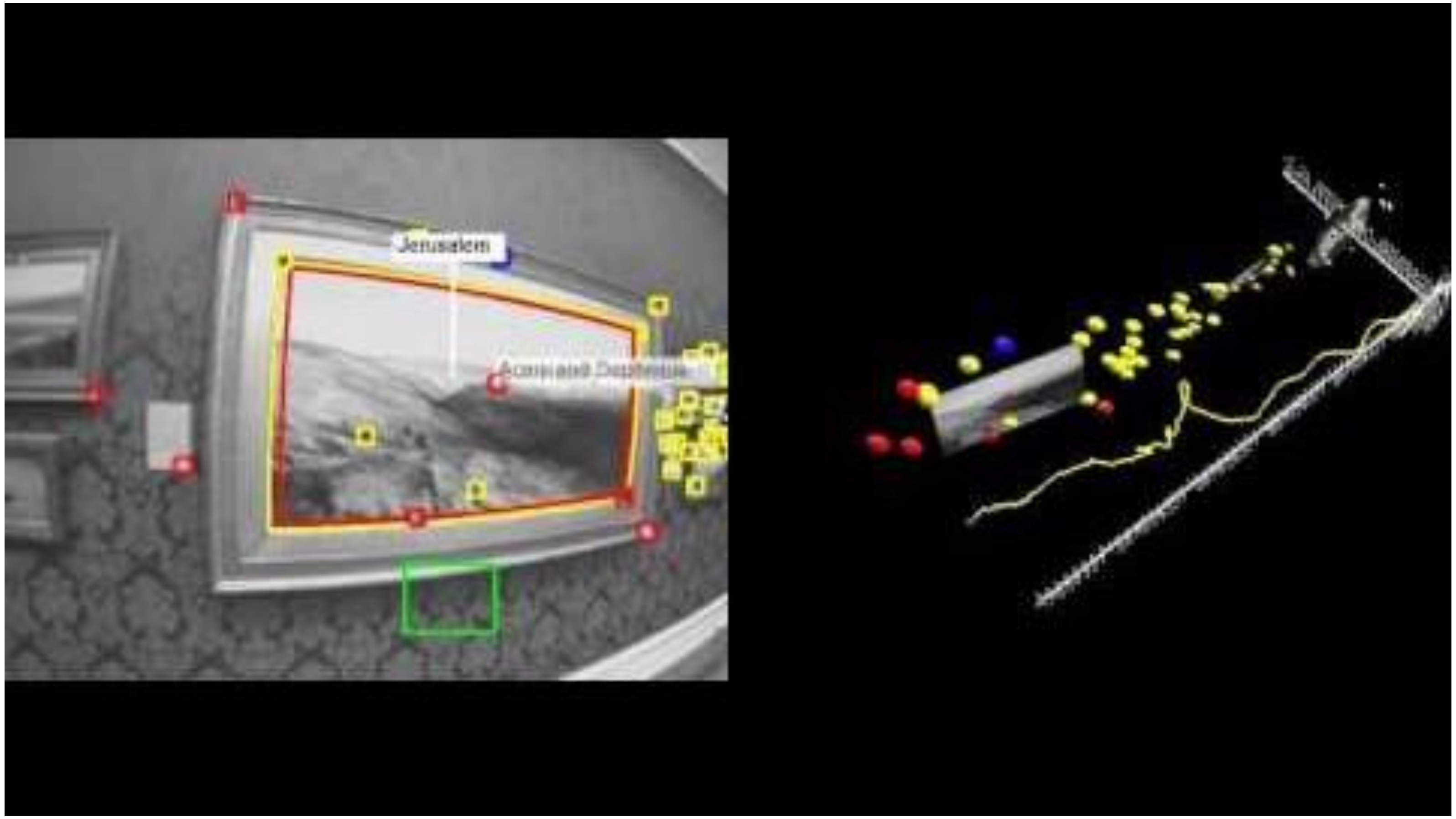
✓ DTAM (Dense Tracking And Mapping, Newcombe, 2011)

- 3D surface reconstruction from every pixel in an image

✓ Deep learning-based SLAM systems (2010s – present)

- Use machine learning algorithms to extract features from sensor data and to build maps

MonoSLAM



https://youtu.be/saUE7JHU3P0

LSD-SLAM (Eagel 2014)

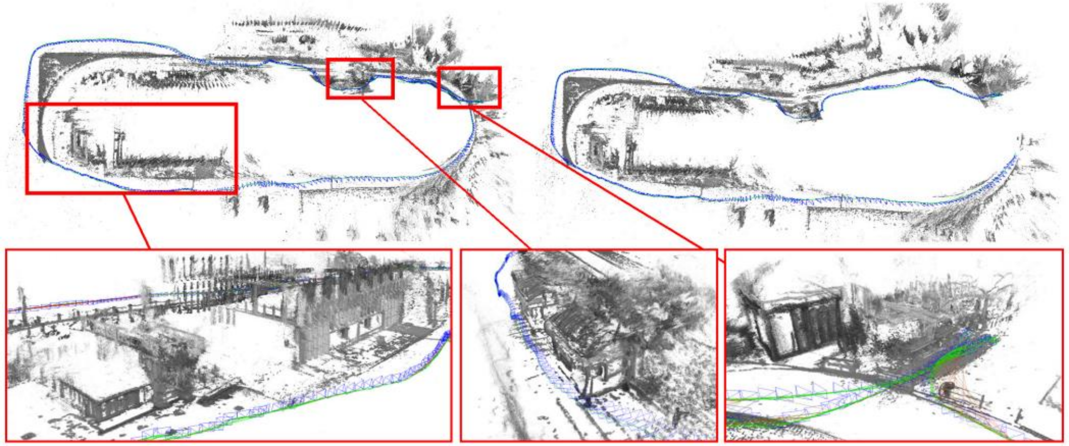
✓Large-Scale Direct monocular SLAM technique

✓Uses image intensities both for tracking and mapping

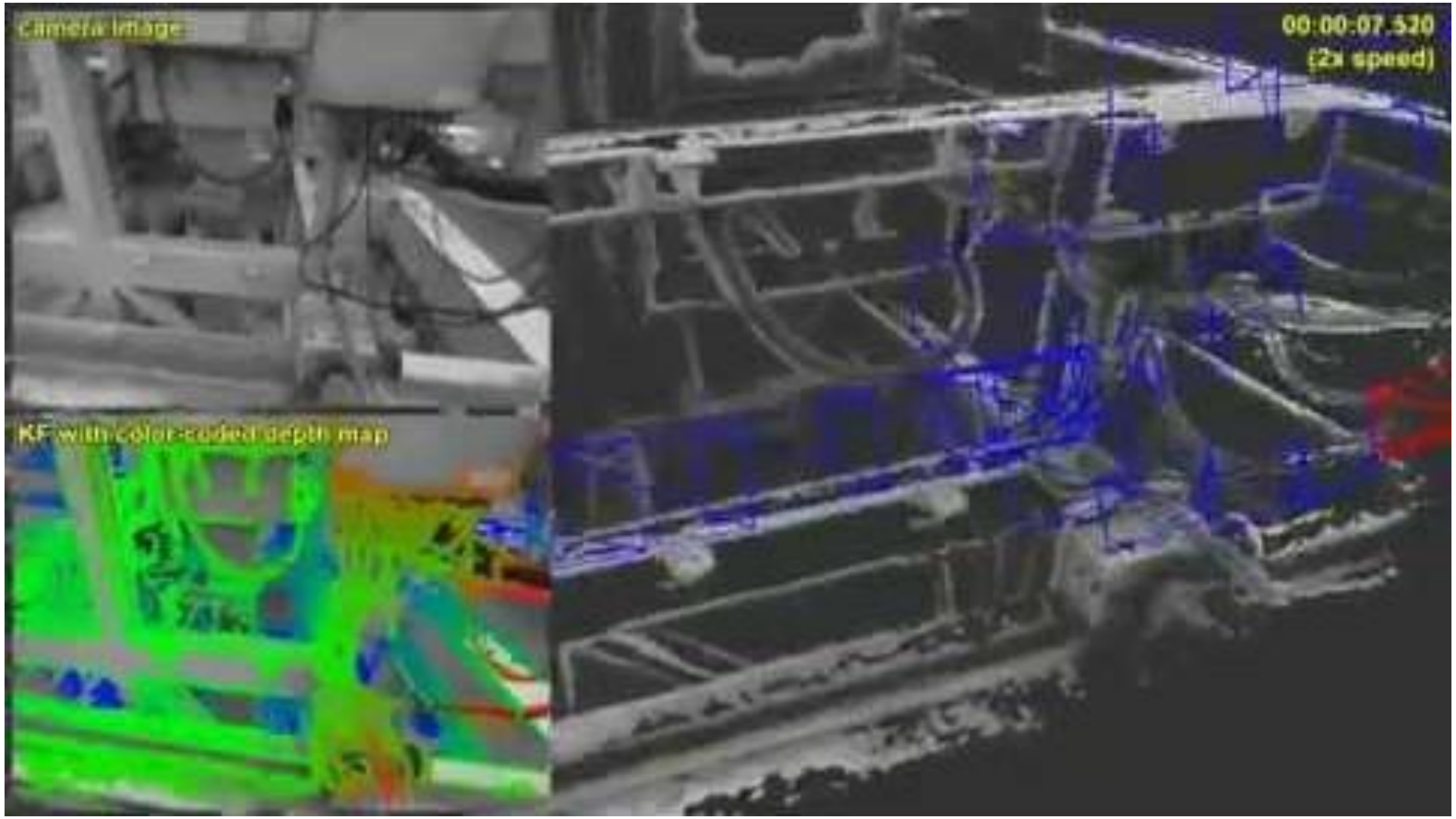
- The camera is tracked using direct image alignment, while geometry is estimated as semi-dense depth maps

✓Support very large-scale tracking

✓Run in real time on CPU and smartphone



LSD-SLAM



https://youtu.be/GnuQzP3gty4

Applications of SLAM Systems

✓Many possible applications

- Augmented Reality camera tracking
- Mobile robot localization
- Real world navigation aid
- 3D scene reconstruction
- 3D object reconstruction
- …

✓Assumptions

- Camera moves through an unchanging scene
- Not suitable for person tracking, gesture recognition

Hybrid Tracking

✓Combining several tracking modalities together

Combining Sensors and Vision

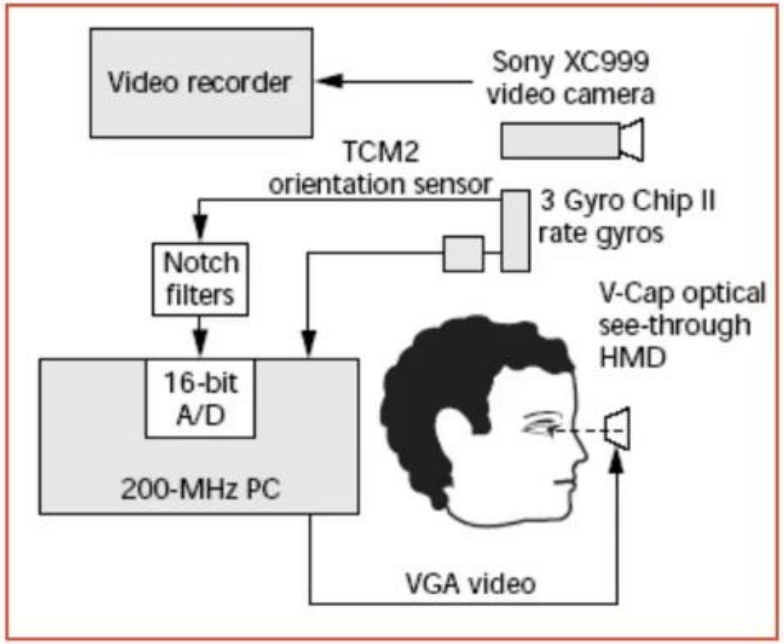
✓ Sensors

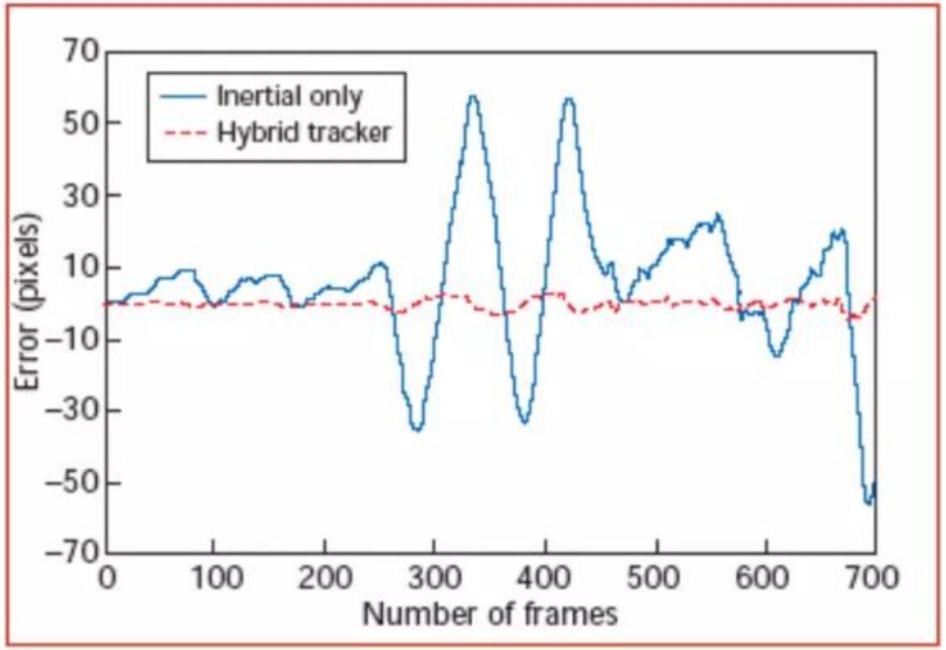
- Produce noisy output (= jittering augmentations)
- Not sufficiently accurate (=wrongly placed augmentations)
- Give the first information on where a user is in the world, and what a user is looking at

✓ Vision

- More accurate (= stable and correct augmentations)
- Require choosing the correct keypoint DB to track from
- Require registering our local coordinate frame (online-generated model) to the global one (world)

Outdoor AR Tracking System





You, Neumann, Azuma outdoor AR system (1999)

Types of Sensor Fusion

✓Complementary

- Combine sensors with different degrees of freedom
- Sensors must by synchronized (or require inter-/extrapolation)
  - E.g., combine position-only and orientation-only sensor
  - E.g., orthogonal 1D sensors in gyro or magnetometer are complementary

✓Competitive

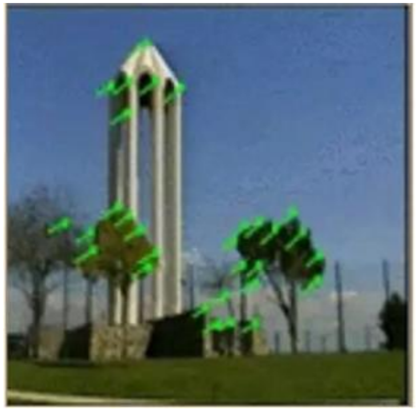
- Different sensor types measure the same degree of freedom
- Redundant sensor fusion: Use worse sensor only if better sensor is unavailable
- Statistical sensor fusion

Example: Outdoor Hybrid Tracking

✓Combine computer vision and inertial gyroscope sensors

✓Both correct for each other

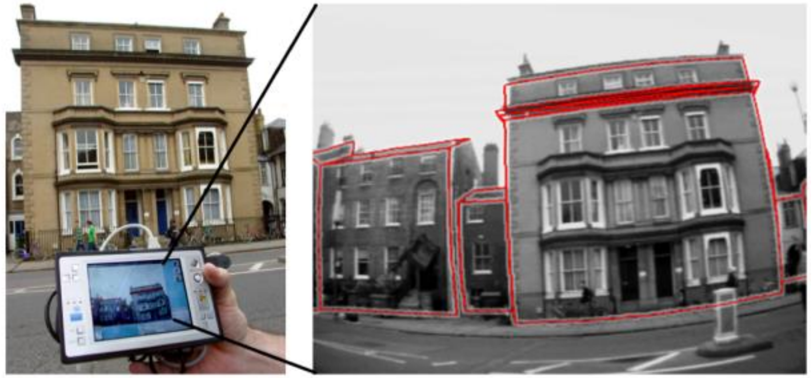
- Inertial gyro
  - Provides frame-to-frame prediction of camera, orientation, fast sensing
  - Drifts over time
- Computer vision
  - Natural feature tracking, corrects for gyro drift
  - Slower, less accurate



Robust Outdoor Tracking

✓Hybrid tracking: computer vision, GPS, inertial

✓Going out, Reitmayr & Drummond (U. Cambridge)

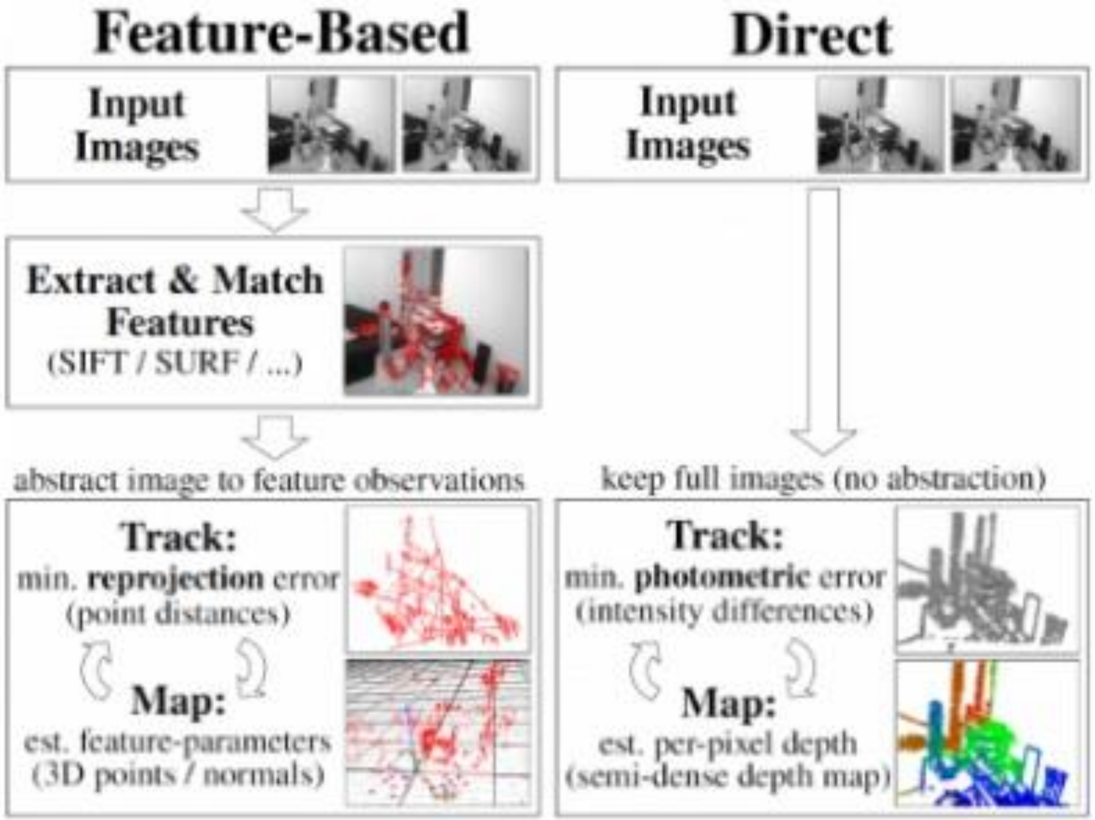




Reitmayr, G., & Drummond, T. W. (2006). Going out: robust model-based tracking for outdoor augmented reality. In Mixed and Augmented Reality, 2006. *ISMAR 2006. IEEE/ACM International Symposium on* (pp. 109-118)

Feature-Based vs. Direct Method

✓ Feature-based approach that only uses small patches around corners and edges

✓Direct uses all information in an image



Feature-Based vs. Direct Method

✓Feature-based

- Can only use & reconstruct corners
- Faster
- Flexible: outliers can be removed retroactively
- Robust to inconsistencies in the model/system
- Decisions based on less complete information
- No need for good initialization

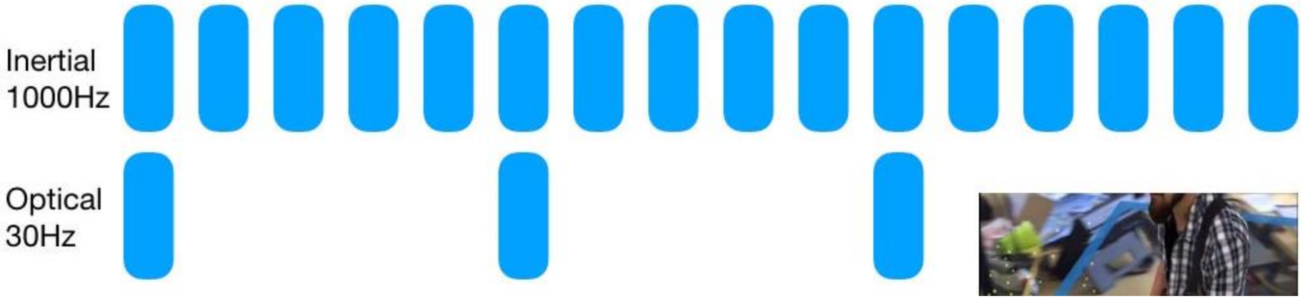
✓Direct

- Can use & reconstruct whole image
- Slower (but good for parallelism)
- Inflexible: difficult to remove outliers retroactively
- Not robust to inconsistencies in the model/system
- Decision based on more complete information
- Need good initialization

ARKit – Visual Inertial Odometry

- ✓ Use both computer vision + inertial sensing
- ✓ Tracking position twice
  - Computer vision – feature tracking, 2D plane tracking
  - Inertial sensing – using the phone IMU
- ✓ Output combined via Kalman filter
  - Determine which output is most accurate
  - Pass pose to ARKit SDK
- ✓ Each system complements the other
  - Computer vision – needs visual features
  - IMU – drifts over time, doesn’t need features

ARKit – Visual Inertial Odometry



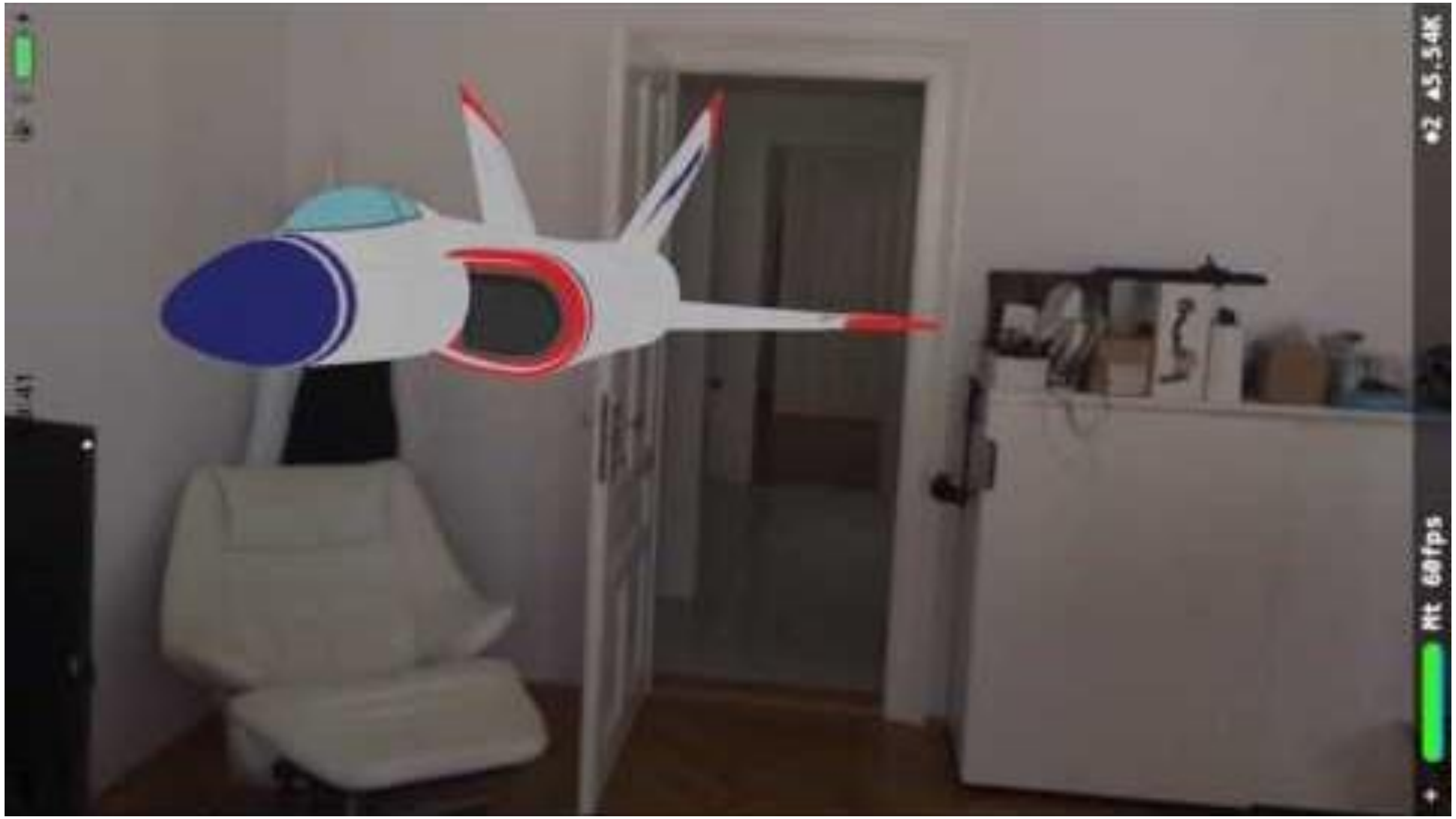
✓Slow camera

✓Fast IMU

✓If the camera drops out, IMU takes over

✓Camera corrects IMU errors

ARKit Demo



https://youtu.be/dMEWp45WAUg

Type	Trigger / tracker	Necessary to generate tracker beforehand	Web support (web AR)	Complexity
Markers	Non-figurative image (marker)	✓	✓	
Images (image tracking)	Figurative Image	✓	✓	
Location-based	GPS Location	✗	✓	
Object tracking	Mesh / 3D Model	✓	✗	
Face tracking	Human face	✗	✓	
Body tracking	Human body, foot, hand	✗	✓	
World tracking	Plano horizontal o vertical	✗	✓	
Spatial tracking	Full-stay scanning (limited environment)	✓	✗	
World mapping	Scanning of large open spaces (world mapping)	✓	✓	

Wrap-up

✓Tracking and registration are key problems

✓Registration error

- Measure against static error
- Measure against dynamic error

✓AR typically requires multiple tracking technologies

- Computer vision most popular

✓Research areas

- SLAM systems, deformable models, mobile outdoor tracking

Q/A