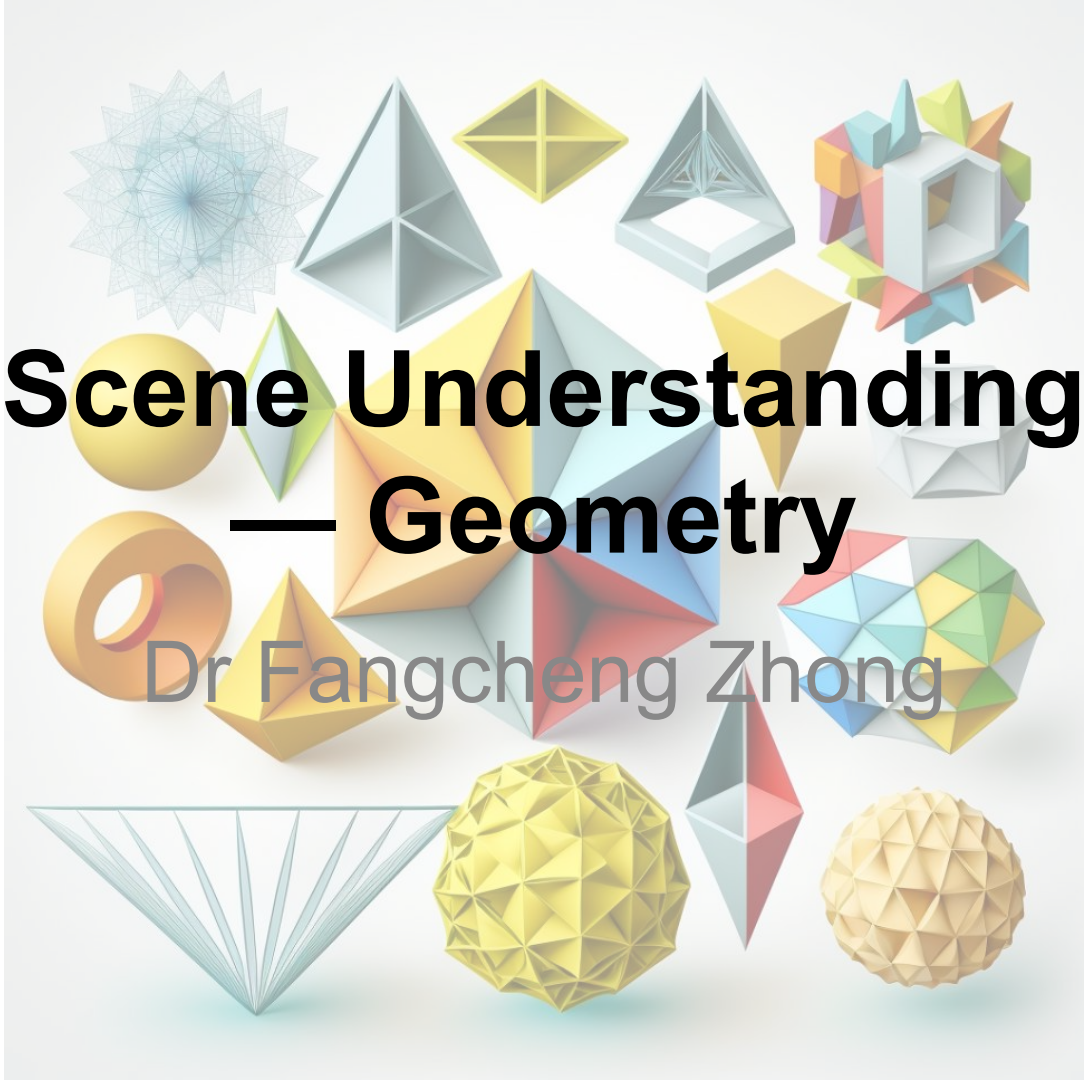


# Scene Understanding — Geometry

Dr Fangcheng Zhong

UNIVERSITY OF CAMBRIDGE

# Last Lecture

- Cross-view correspondence is crucial for motion tracking and 3D mapping
- Solution: feature tracking

# Outline

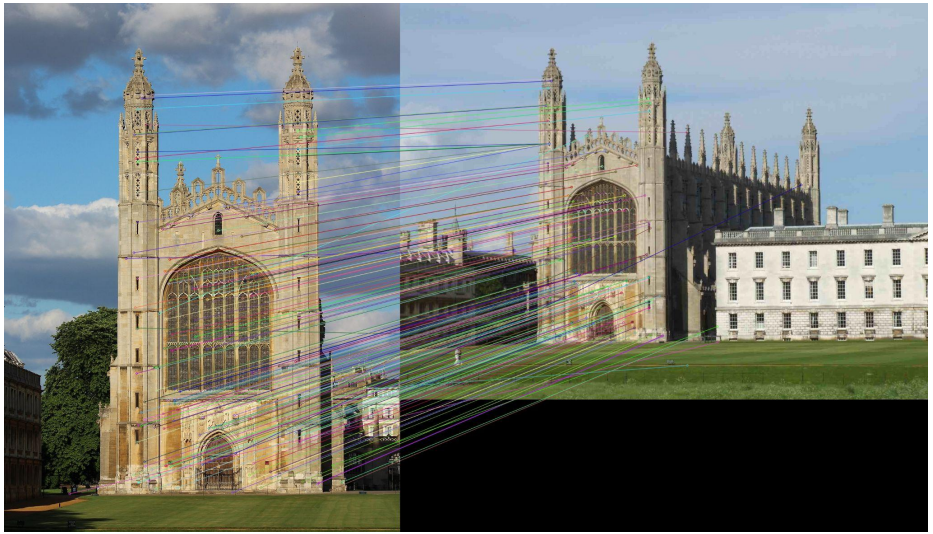
- Feature tracking
  - detection, description, matching
  - SIFT, SURF, ORB
  - RANSAC
- Efficient geometric representations
  - featured point clouds
  - plane tracking
  - depth image
- Object tracking
  - face/body/gesture tracking

# Feature Tracking

**detection**

**description**

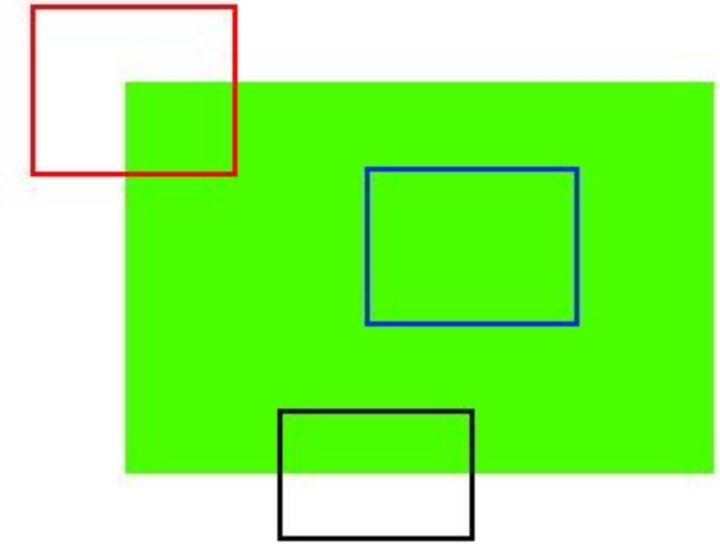
**matching**



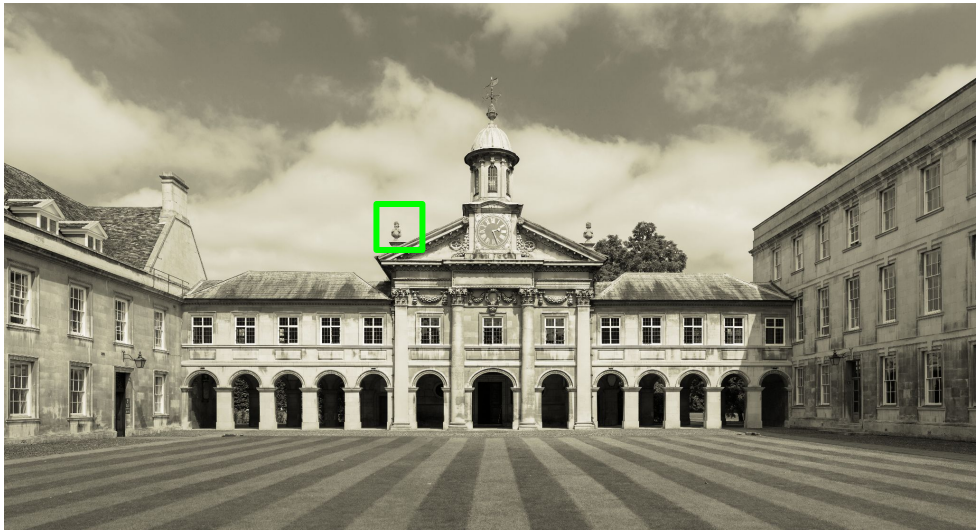
# Feature Detection

# Feature Detection

- Textureless regions can result in failure of motion tracking, 3D mapping, and related processing (hit test, occlusion, tracking)



# Harris Corner Detection



- Weighted sum of second moment matrices over a window for each pixel

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

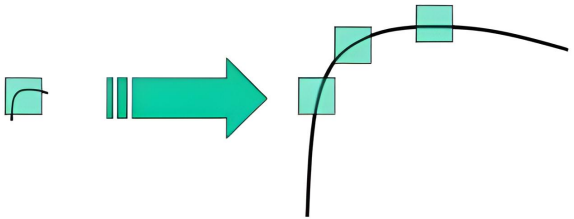
- Eigenvalues of **M** indicates whether the window represents a corner
- Harris corner response function

$$R = \det(M) - \alpha \,\mathrm{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

- Non-maximum suppression to refine the results

# Harris Corner Detection

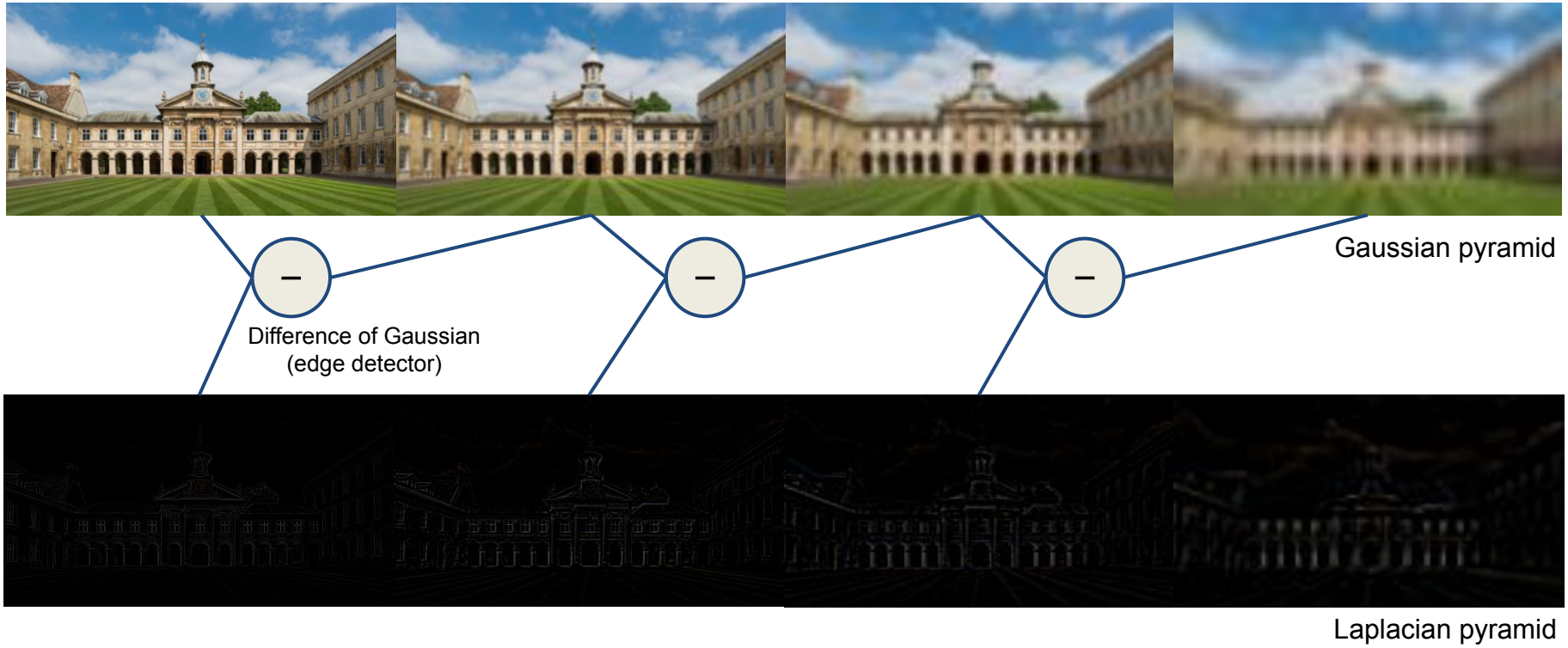
- ✅
  - Rotation invariant
  - Invariant to additive (exposure) and multiplicative (contrast) changes
- ❌
  - Not invariant to scaling



# SIFT Keypoint Detection

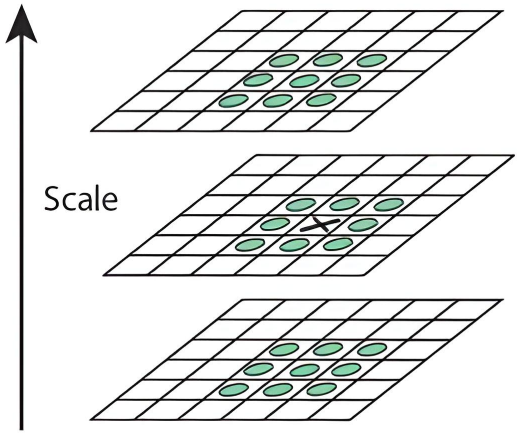
- Scale-Invariant Feature Transform
  - find the correct scale of the keypoint via Laplacian pyramids

# SIFT Keypoint Detection



# SIFT Keypoint Detection

- Search for key points from local extrema over scale and space
- Remove edge points
- Contrast threshold to refine the results



# SIFT Keypoint Detection

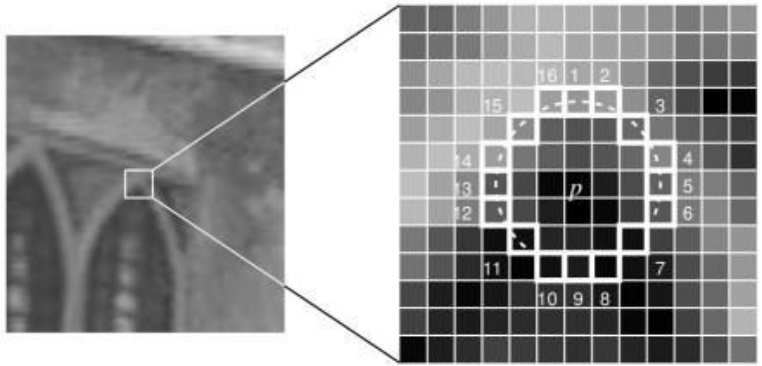
- ✅
  - Invariant to a wide range of geometric transformations, including scale, rotation, and affine distortion
- ❌
  - Computationally expensive on large images

# SURF Keypoint Detection

- Speeded-Up Robust Features
  - Approximate Difference of Gaussian with box filtering
  - Fast to compute with integral images
  - Can be computed in parallel for different scales
- ✅
  - Computationally efficient, well-suited for real-time applications
- ❌
  - Less invariant to rotations and affine distortions

# FAST Algorithm

- Features from Accelerated Segment Test
  - Consider a circle of 16 pixels around a pixel **p**
  - **p** is a corner if there exists a set of 12 contiguous pixels in the circle which are all brighter (darker) than I_p + t (I_p − t)
  - High-speed test with machine learning
  - Non-maximal suppression to refine the results

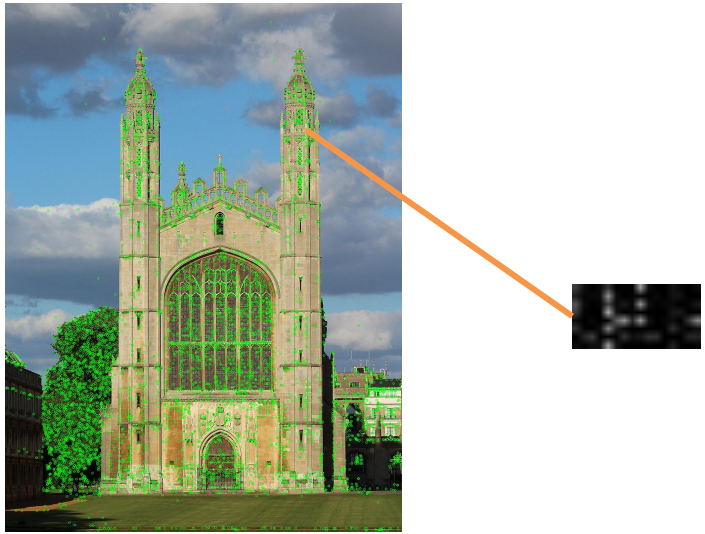


# FAST Algorithm

- ✅
  - Faster than other existing corner detectors
- ❌
  - Not robust to high levels of noise and texture
  - Not robust to rotation and scaling

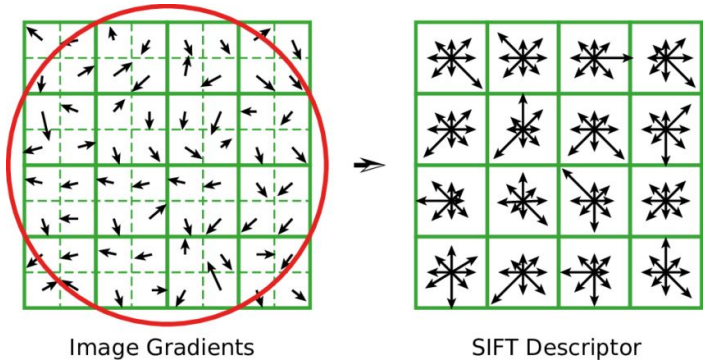
# Feature Descriptors

- Encode each detected keypoint into a feature vector for matching



# SIFT Feature Descriptor

- Identify the principal orientation (from dominate gradient) and scale (from the pyramid) of the keypoint
- Rotate and scale the local patch of the keypoint accordingly
- Divide each patch into 4x4 subpatches and generate an 8-bin gradient histogram for each subpatch
- The descriptor is a 4x4x8=128 vector

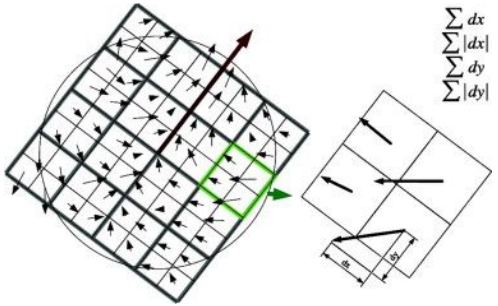


# SIFT Feature Descriptor

- ✅
  - Robust to a wide range of transformations
  - Highly discriminative features, accurate for matching
- ❌
  - Intensive in both memory and computation

# SURF Feature Descriptor

- Use wavelet responses in horizontal and vertical direction for orientation assignment
- Divide each patch into 4x4 subpatches and generate a 4d horizontal and vertical wavelet responses
- The descriptor is a 4x4x4=64 vector

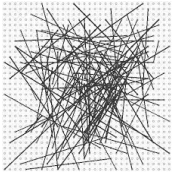


# SURF Feature Descriptor

- ✅
  - Faster matching (use of sign of Laplacian)
  - Robust to blurring and rotation
- ❌
  - Not robust to viewpoint change and illumination change

# BRIEF Feature Descriptor

- Binary Robust Independent Elementary Features
  - small binary strings that are easy to compute and compare
- Procedure
  - select a patch around a keypoint
  - select a set of pixel pairs in that patch



  - for each pair, generate a binary number comparing pixel intensities

$$b = \begin{cases} 1 & \text{if } I(s_1) < I(s_2) \\ 0 & \text{otherwise} \end{cases}$$

  - concatenate all the binaries to a bit string

# BRIEF Feature Descriptor

- ✅
  - Compact descriptor (binary, short length)
  - Fast to compute (simple comparison)
  - Fast to compare (hamming distance)
- ❌
  - Reduced accuracy
  - Not robust to rotation and scale
  - Not robust to high levels of noise

# Oriented FAST and Rotated BRIEF (ORB)

- Fusion of FAST keypoint detector and BRIEF descriptor with modifications
  - Compute multiscale-features and orientation of keypoints
  - Add rotation compensation
  - Learn optimal sampling pairs

# Feature Tracking

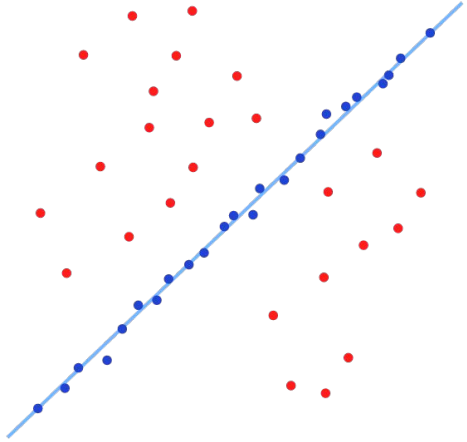
- Detector
  - Repeatability across multiple images (despite geometric and photometric transformations)
  - Precision & locality (occupies small an area of the image, robust to clutter and occlusion)
- Descriptor
  - Saliency & matchability (distinctive description, correspondence despite geometric and photometric distortions)
  - Compactness and efficiency

# Feature matching

- Threshold distance for matching
- Higher threshold gives more good or bad pairs
- Fast matching algorithms
  - kd tree
  - k-nearest neighbors
  - Fast Library for Approximate Nearest Neighbors (FLANN)

# Random Sample Consensus (RANSAC)

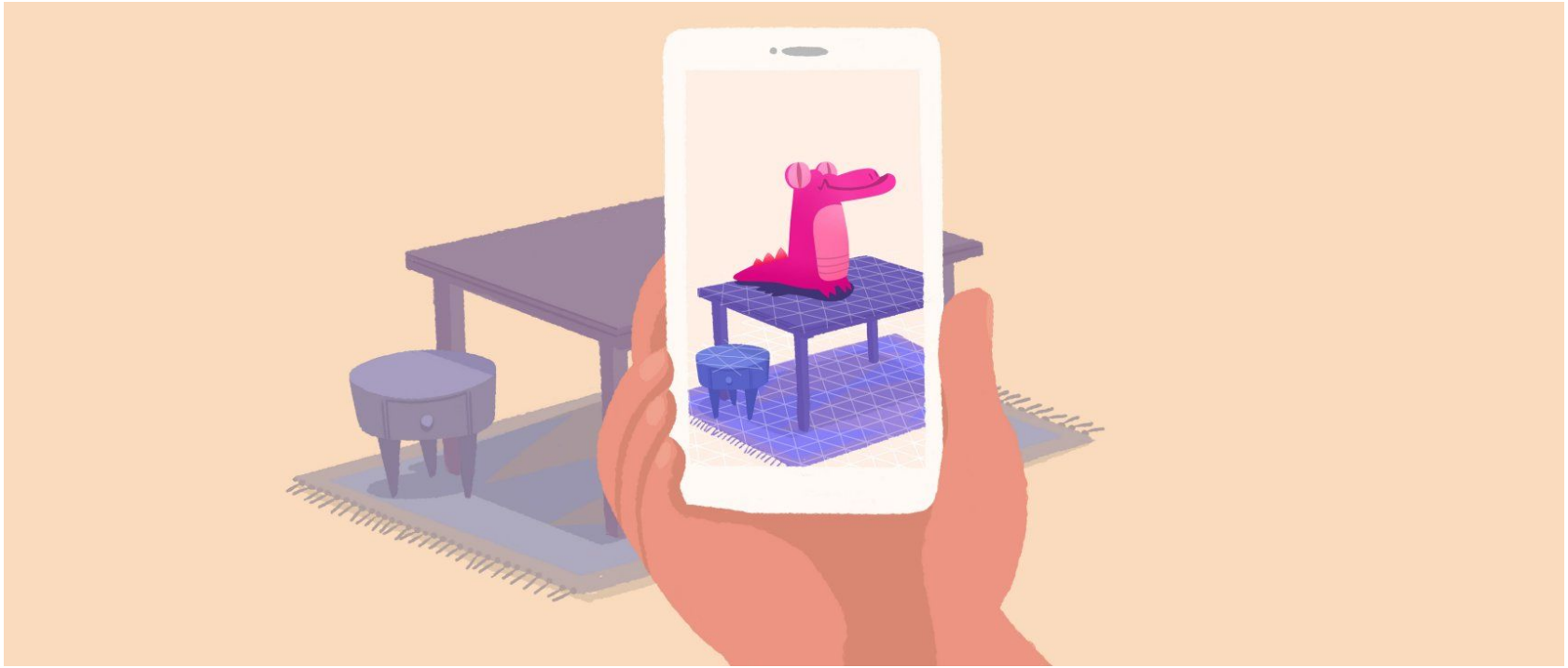
- An iterative method for estimating a mathematical model from a data set that contains outliers
- Reject outliers (incorrect correspondence) in motion tracking and 3D mapping



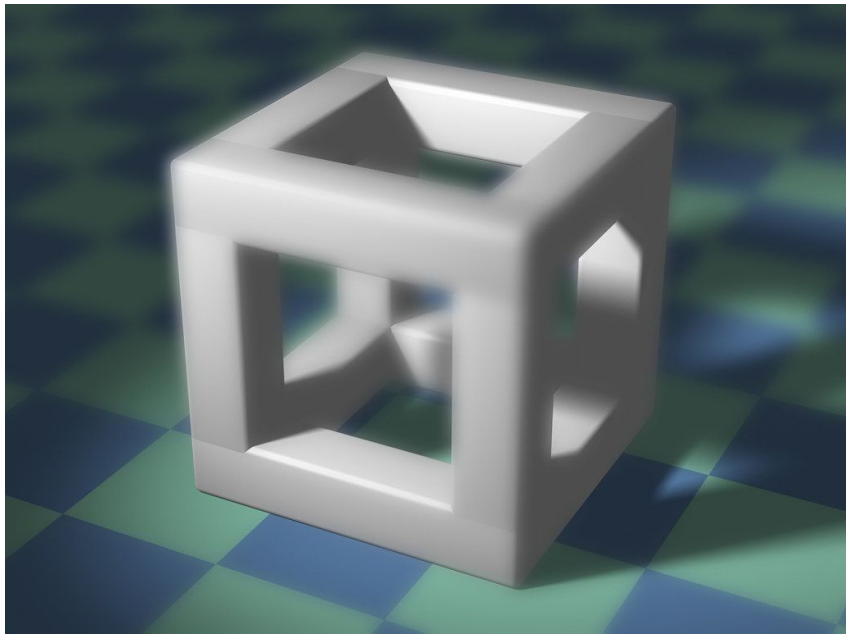
# Featured Point Clouds

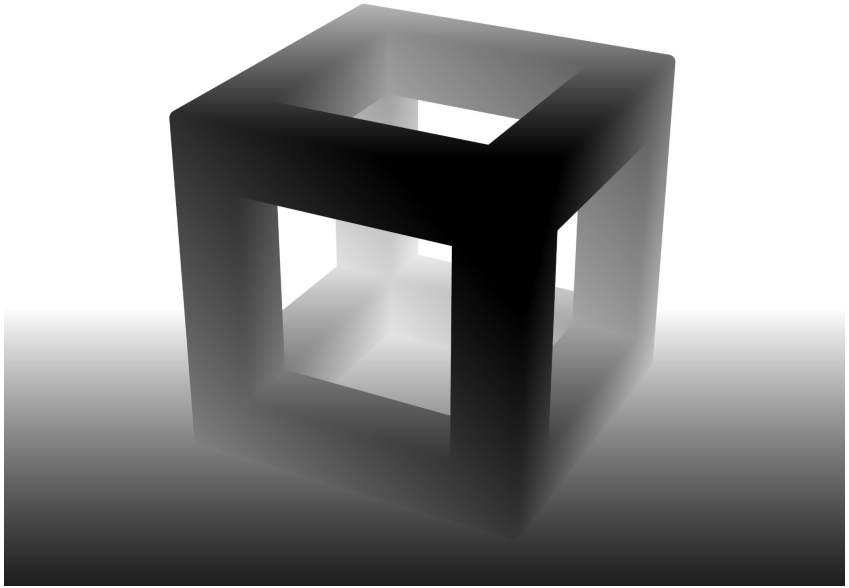
- 3D position
- feature vector
- normal

# Plane Detection

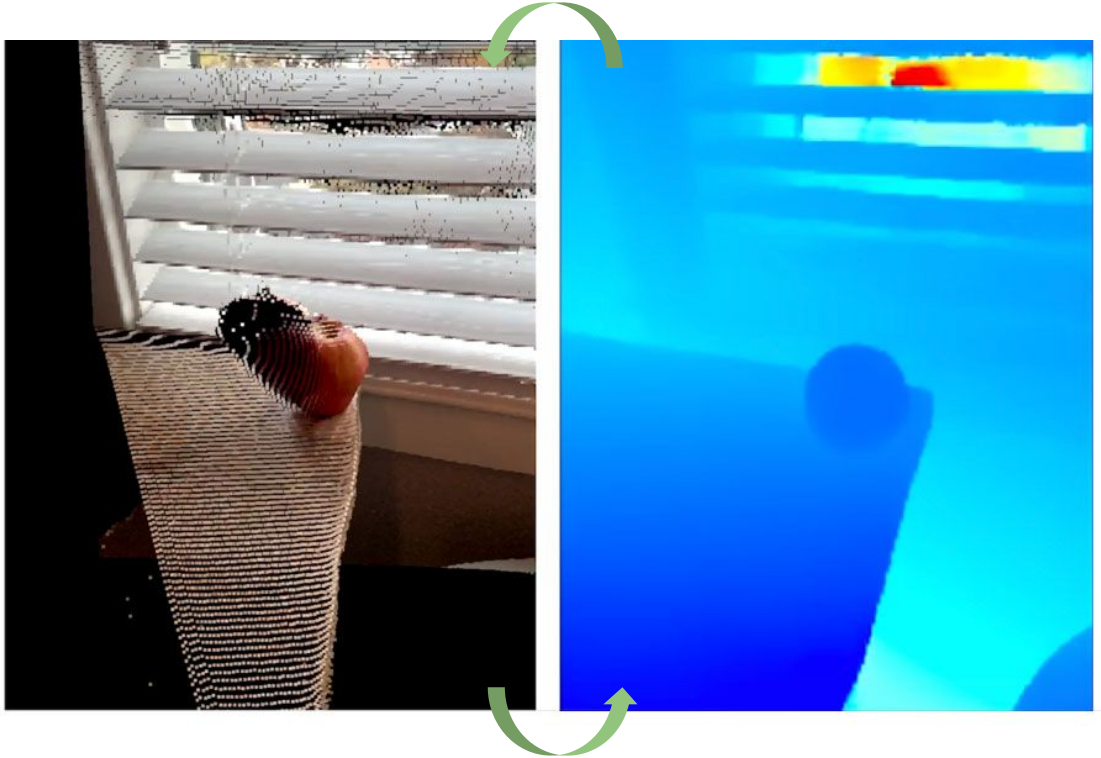


# Depth Image

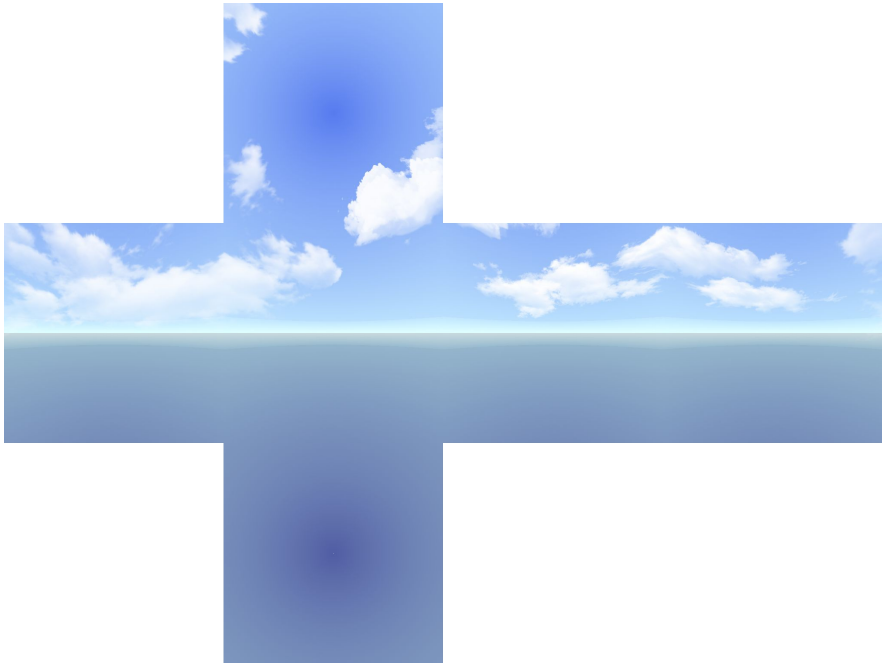


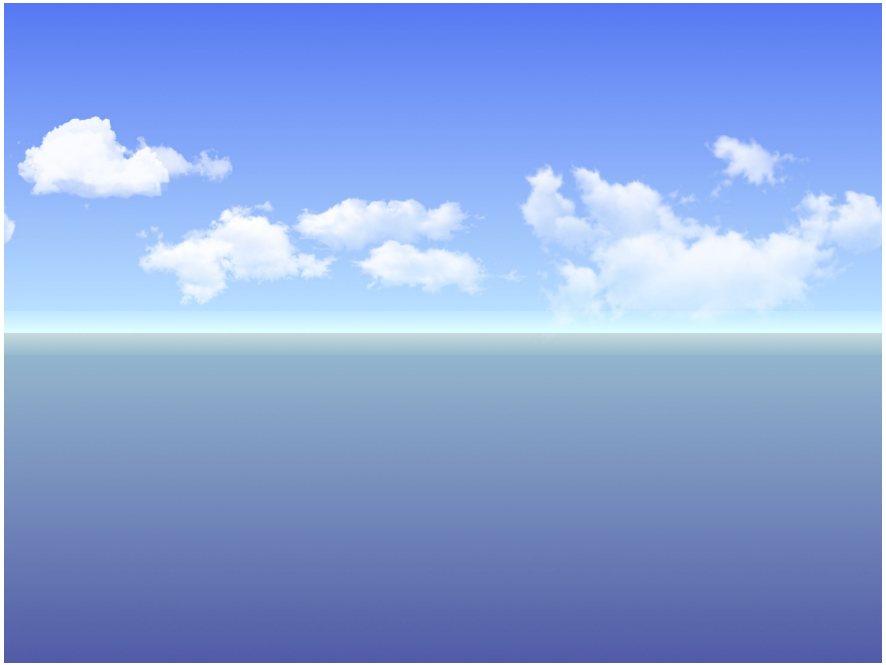


# Depth Image

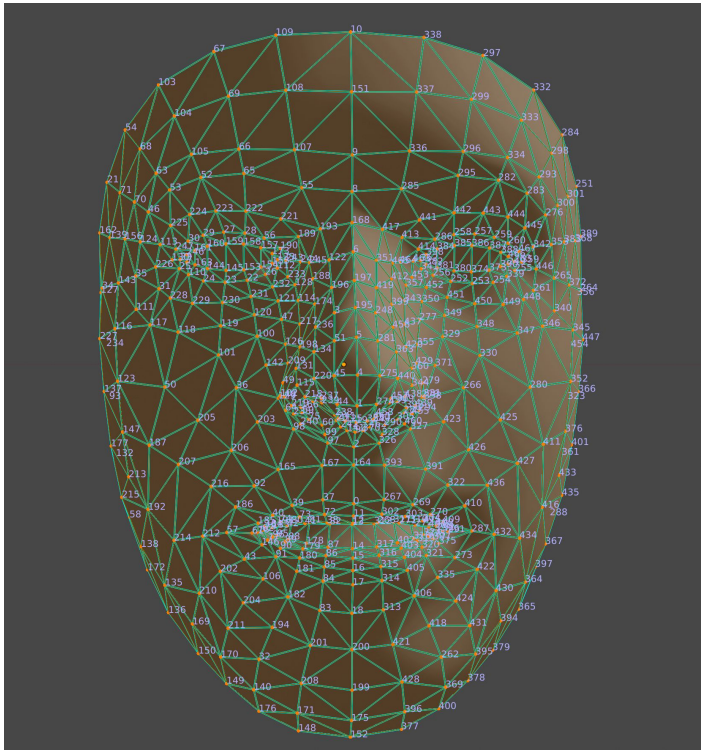


# Skybox Image





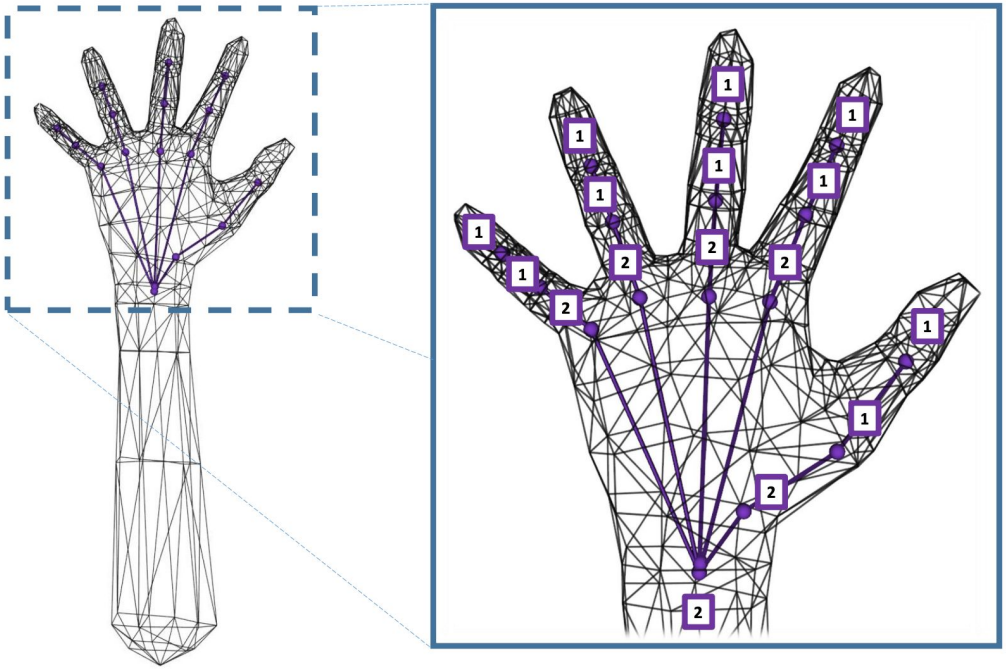
# Face Tracking



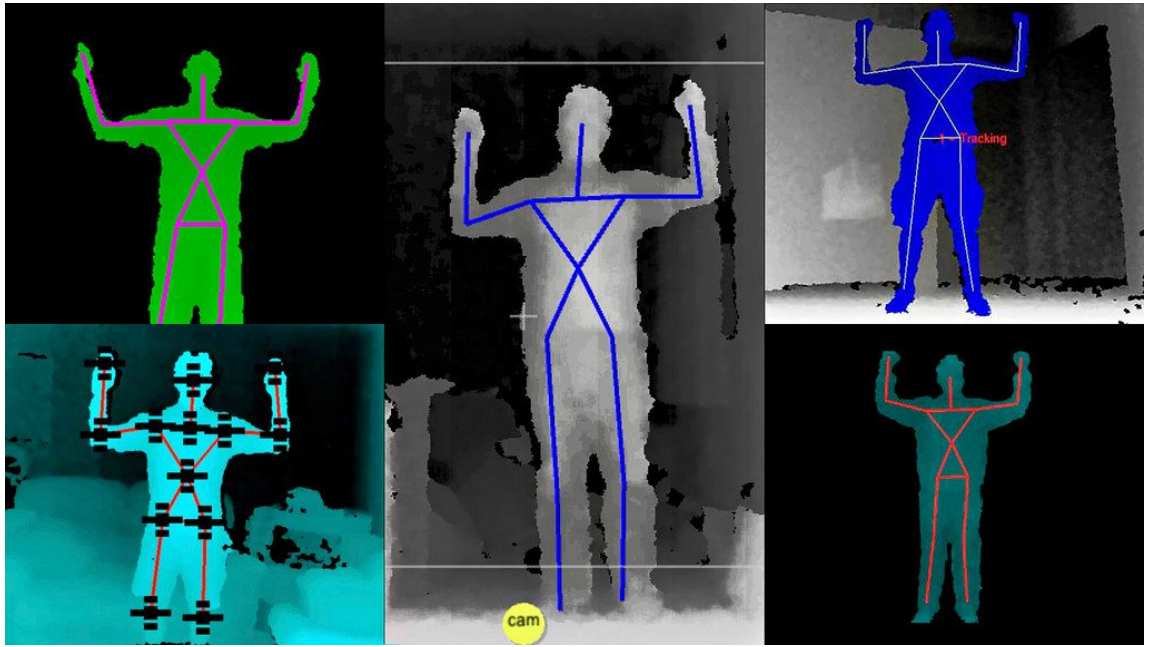
- keypoints detection
- head pose estimation
- parametric face model (e.g. 3DMM)

# Face Tracking

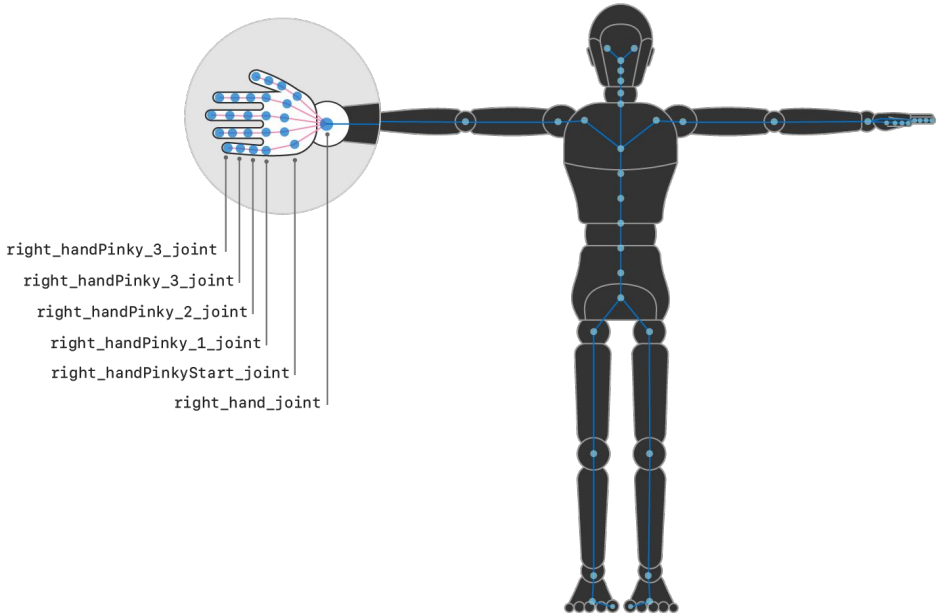
# Hand Pose Tracking

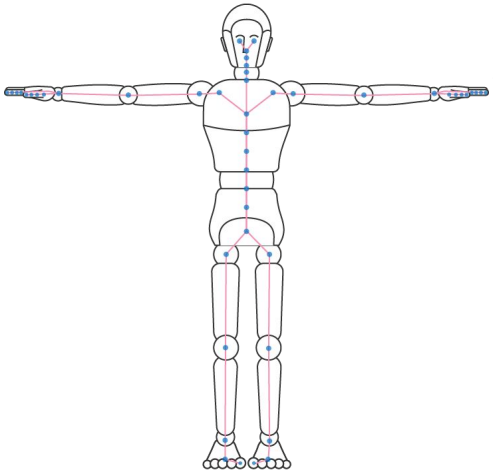


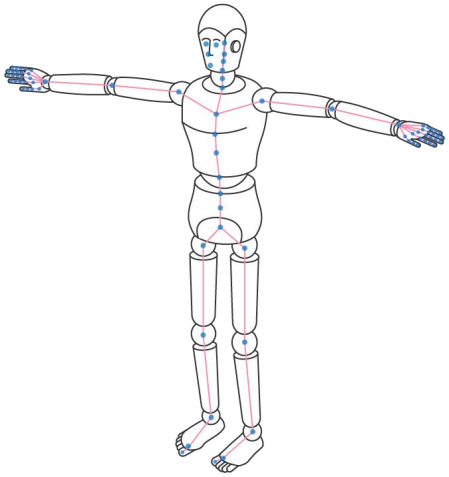
# Body Pose Tracking




# Body Pose Tracking

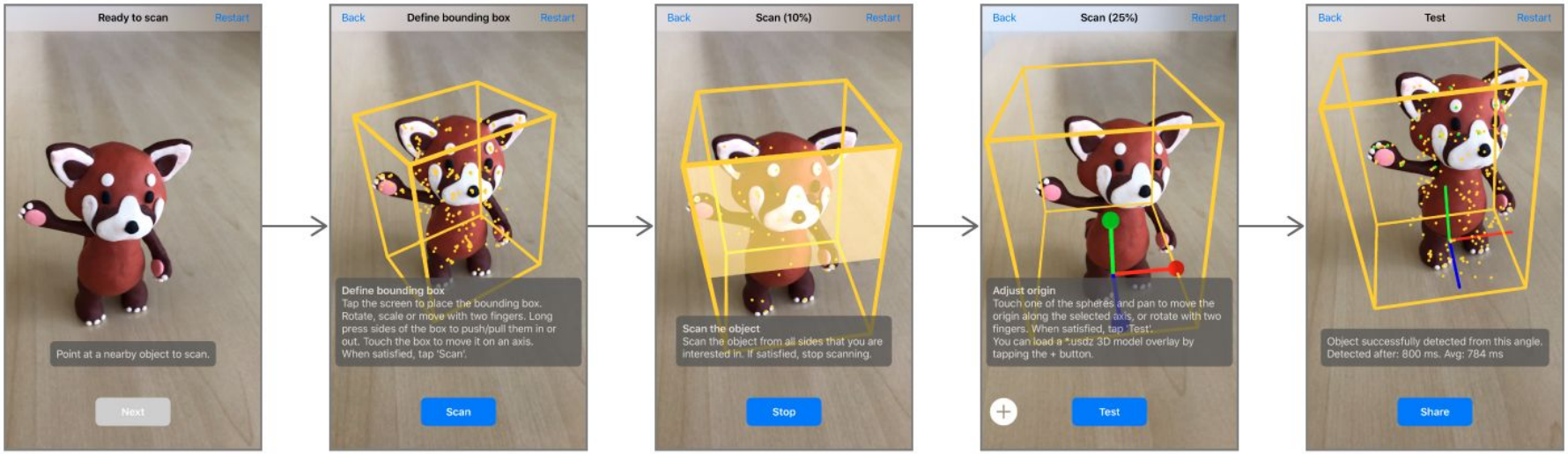




**ARKit body pose tracking**

# Object Tracking




Prepare to scan | Define bounding box | Scan | Adjust origin | Test and Export

**ARKit object tracking**