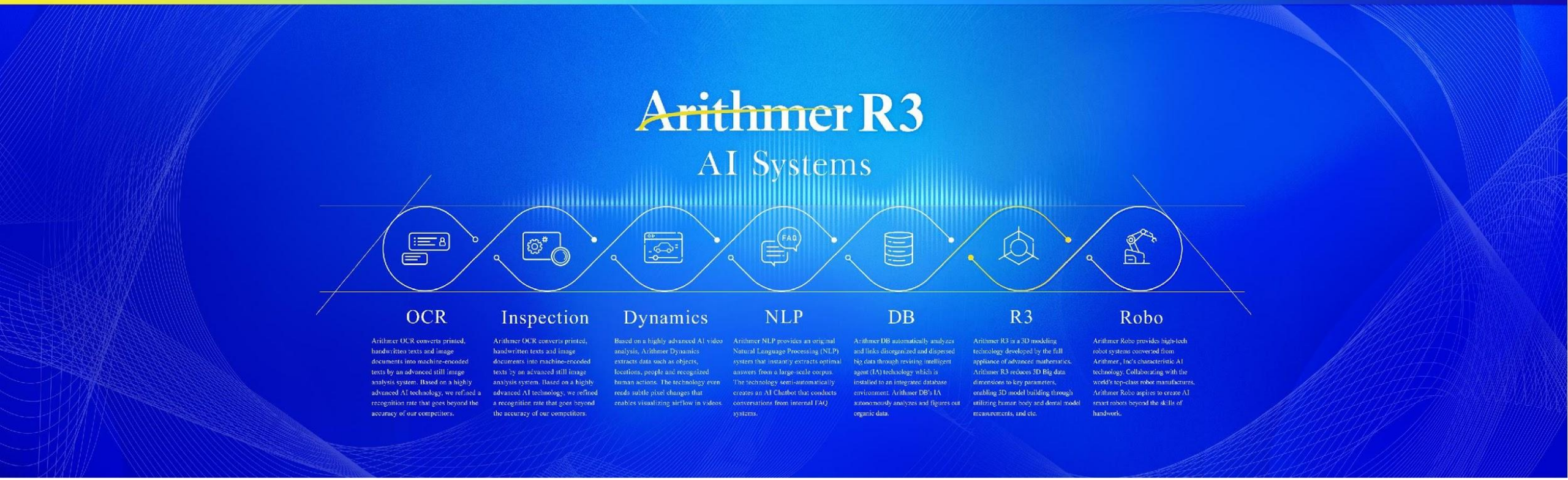


Object Pose Estimation

Arithmer R3 Div. R3 - Ryosuke Sasaki

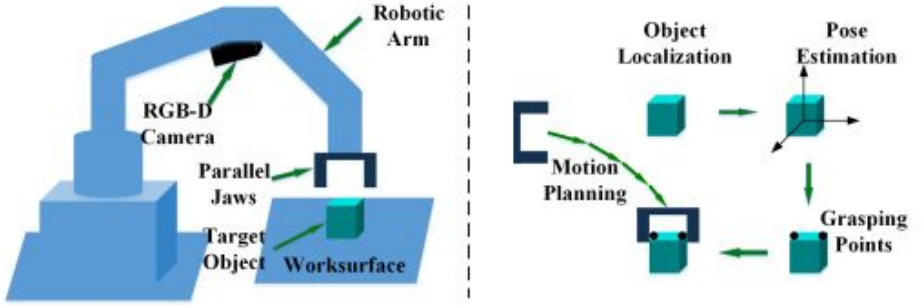


Figure 1: The grasp detection system. (Left)The robotic arm equipped with the RGB-D camera and two parallel jaws, is to grasp the target object placed on a planar worksurface. (Right)General procedures of robotic grasping involves object localization, pose estimation, grasping points detection and motion planning.

- Correspondence-based method : Find Correspondences between 2D points and 3D points (PnP, ICP)
- Template-based method: Extract the gradient information for matching, and refined(ICP)
- Voting-based method: Every local predicts a result, and refined using RANSAC
- Regression-based method: Represent pose suitable for CNN

Table 2: Summary of 6D pose estimation methods.

Method	Known Info	Key idea	Representative methods
Correspondence-based method	3D point cloud, Rendered images from 3D model with texture	Find Correspondences between 2D points and 3D points, and use PNP methods	SIFT [Lowe, 1999], SURF [Bay *et al.*, 2006], ORB [Mur-Artal *et al.*, 2015]
	3D point cloud	Find 3D Correspondence through random hypothesis or 3D descriptors, and results are refined using ICP	Spin image [Johnson, 1997], FPFH [Rusu *et al.*, 2009], SHOT [Salti *et al.*, 2014]
Template-based method	3D point cloud, Rendered images from 3D model with no texture	Extract the gradient information for matching, and results are refined using ICP	LineMod [Hinterstoisser *et al.*, 2012], Hodaň et al. [Hodaň *et al.*, 2015]
Voting-based method	3D point cloud or rendered RGB-D images with pose	Every local predicts a result, and results are refined using RANSAC	Brachmann et al. [Brachmann *et al.*, 2014], PPF [Drost and Ilic, 2012], DenseFusion [Wang *et al.*, 2019a]
Regression-based method	3D point cloud or rendered RGB-D images with pose	Represent pose suitable for CNN	BB8 [Rad and Lepetit, 2017], SSD6D [Kehl *et al.*, 2017], PoseCNN [Xiang *et al.*, 2017], Deep6DPose [Do *et al.*, 2018]

Arxiv:1905.06658

Correspondence-based method

With rich textures to get image features

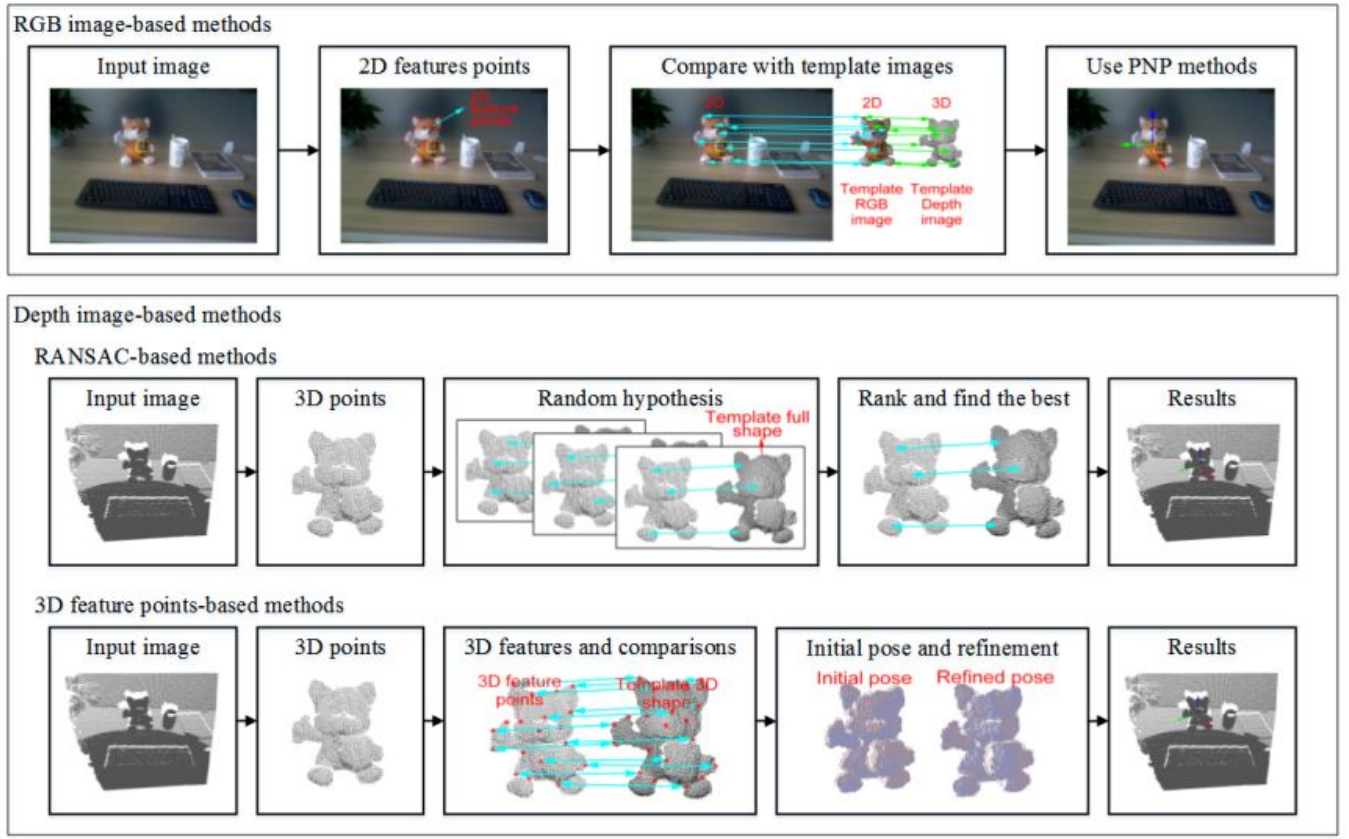


Figure 12: Typical functional flow-chart of correspondence-based object pose estimation methods without object detection.

Arxiv:1905.06658

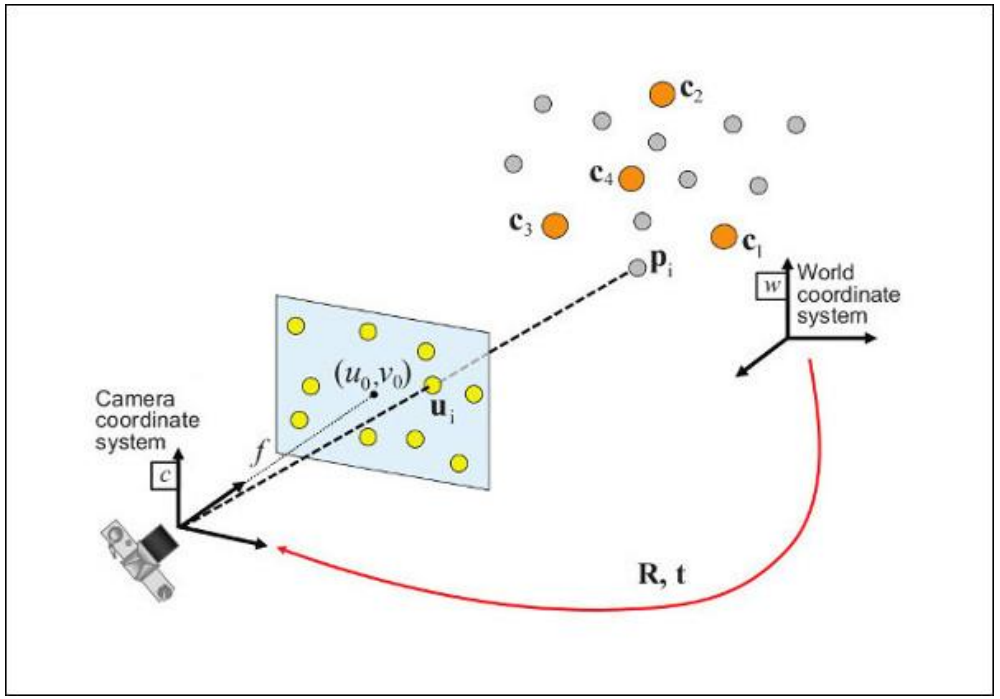
PnP solver

Estimate extrinsic camera parameter using known 3D points in a world coordinate

There are many many solvers (e.g. P3P, P5P.., using RANSAC or Eigen value decomposition)

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- (X, Y, Z) are the coordinates of a 3D point in the world coordinate space
- (u, v) are the coordinates of the projection point in pixels
- A is a camera matrix, or a matrix of intrinsic parameters
- (cx, cy) is a principal point that is usually at the image center
- fx, fy are the focal lengths expressed in pixel units.

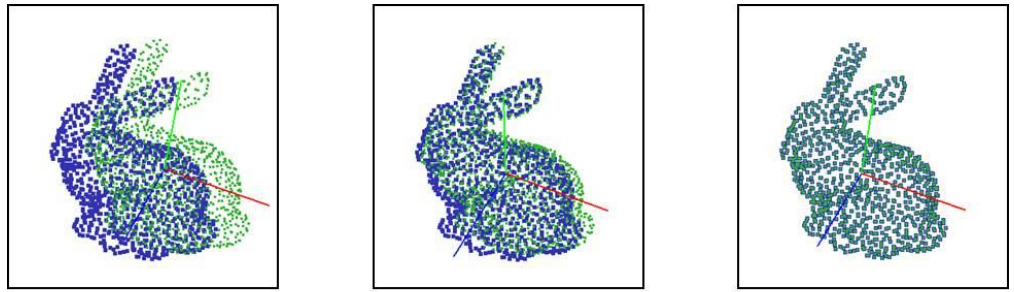


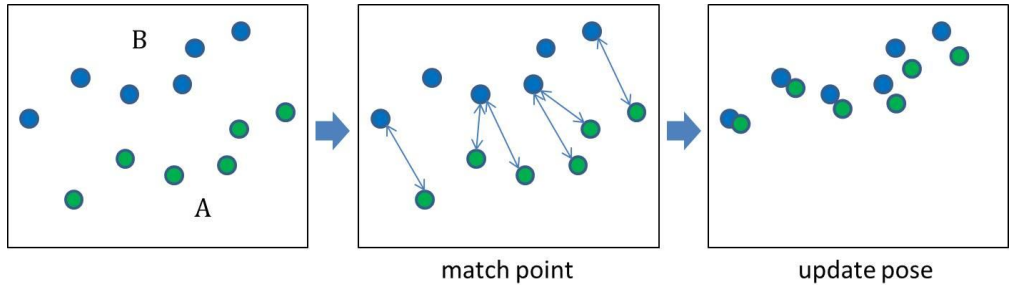
Arxiv:1905.06658

ICP algorithm

Iterative matching for 2 point clouds

1. Correspondence nearest neighbor points
2. Translate a point cloud set to match the other sets
3. iteration above





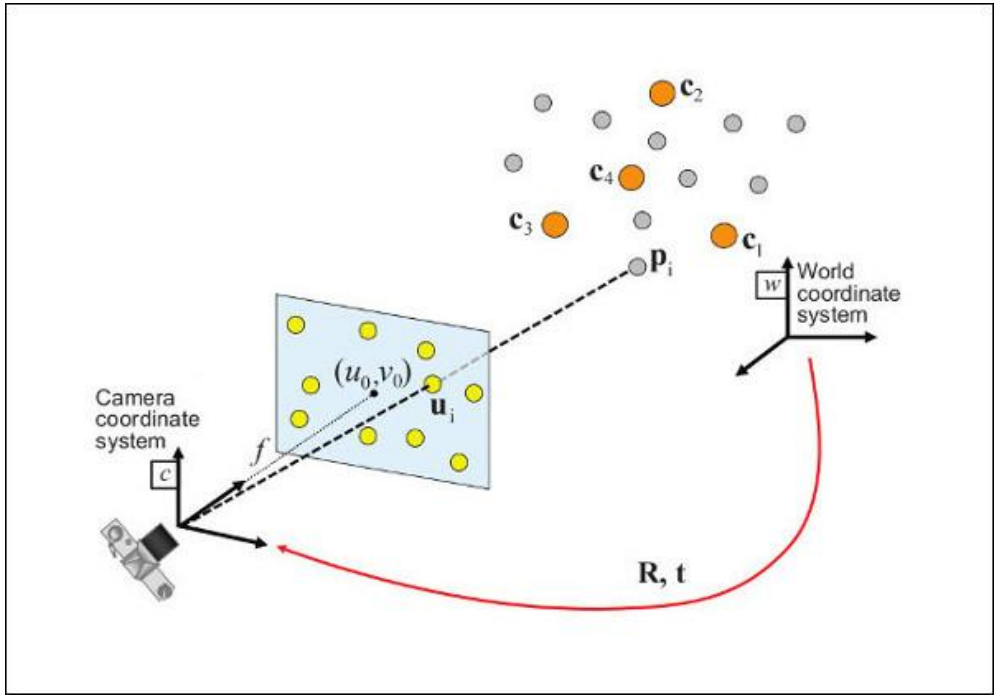
ICP

Estimate extrinsic camera parameter using known 3D points in a world coordinate

There are many many solvers (e.g. P3P, P5P.., using RANSAC or Eigen value decomposition)

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- (X, Y, Z) are the coordinates of a 3D point in the world coordinate space
- (u, v) are the coordinates of the projection point in pixels
- A is a camera matrix, or a matrix of intrinsic parameters
- (cx, cy) is a principal point that is usually at the image center
- fx, fy are the focal lengths expressed in pixel units.



Arxiv:1905.06658

SegICP

Segmentation and depth image-based partial Registration to obtain object's pose

SegNet

Multi-hypothesis object pose:
Point-to-point ICP

1cm pose error and < 5°error

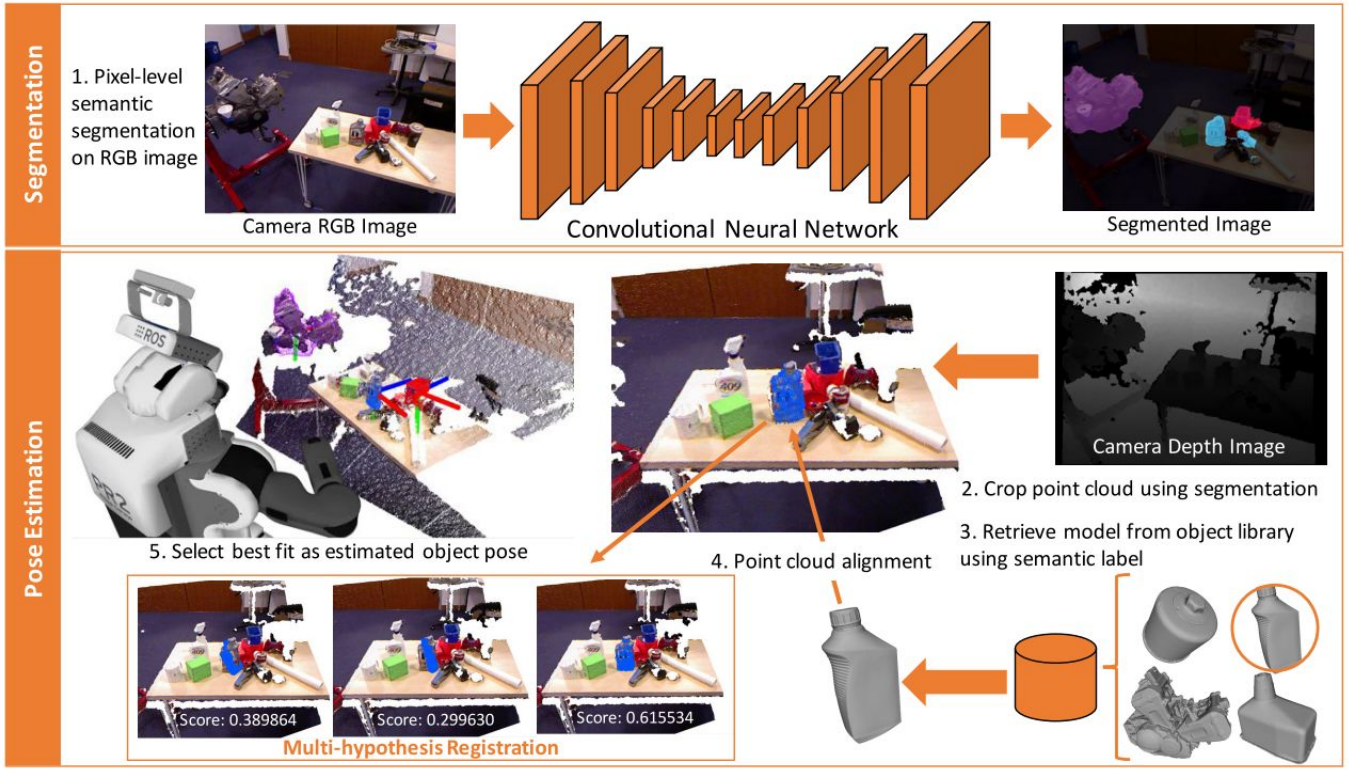


Fig. 2: The full SegICP pipeline operating in a cluttered environment. The system detects objects relevant to the automotive oil change task and estimates a 6-DOF pose for each object. The colored overlay pixels in the segmented image (top right) correspond to a blue funnel (red), an oil bottle (blue), and the engine (purple), as detected by the Kinect1 mounted on top of a PR2 robot. Selected multi-hypothesis registrations for the oil bottle object (bottom left) are shown with their respective alignment scores. The hypothesis registrations are evaluated in parallel to determine the optimal object pose.

Template-based method

Using gradient information in RGB-D or RGB image
Without rich texture

ICP

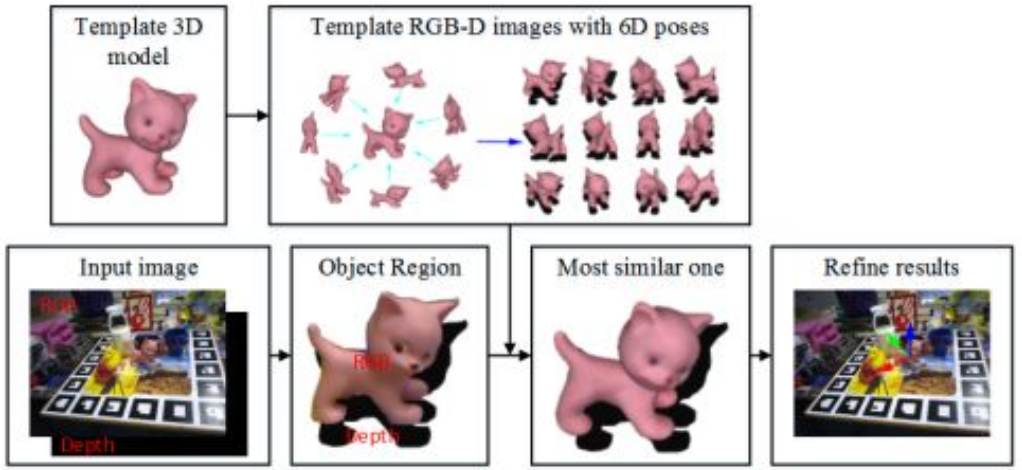


Figure 15: Typical functional flow-chart of template-based object pose estimation methods without object detection. Data from the lineMod dataset [Hinterstoisser *et al.*, 2012].

Figure 16: Template generation and the feature points. Left: Red vertices represent the virtual camera centers used to generate templates. Middle: The color gradient features are displayed in red and surface normal features in green. Right: 15 different texture-less 3D objects used. (Courtesy of [Hinterstoisser *et al.*, 2012])

Arxiv:1905.06658

Voting-based method

Objects with occlusions
Dense class labeling

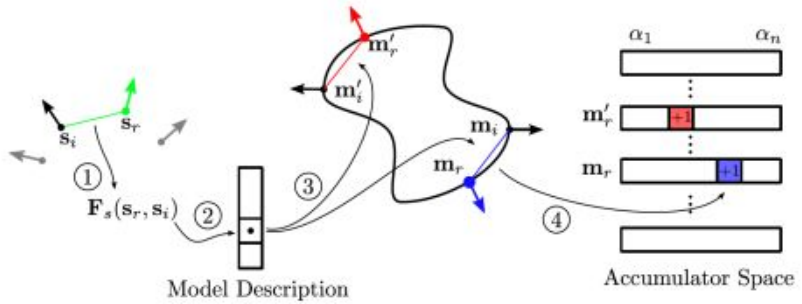


Figure 18: Visualisation of different steps in the voting scheme: (1) Reference point s_r is paired with every other point s_i, and their point pair feature F is calculated. (2) Feature F is matched to the global model description, which returns a set of point pairs on the model that have similar distance and orientation. (3) For each point pair on the model matched to the point pair in the scene, the local coordinate α is calculated by solving $s_i = T_{s\to g}^{-1} R_x(\alpha) T_{s\to g} m_i$. (4) After α is calculated, a vote is cast for the local coordinate (m_i, α). (Courtesy of [Drost *et al.*, 2010])

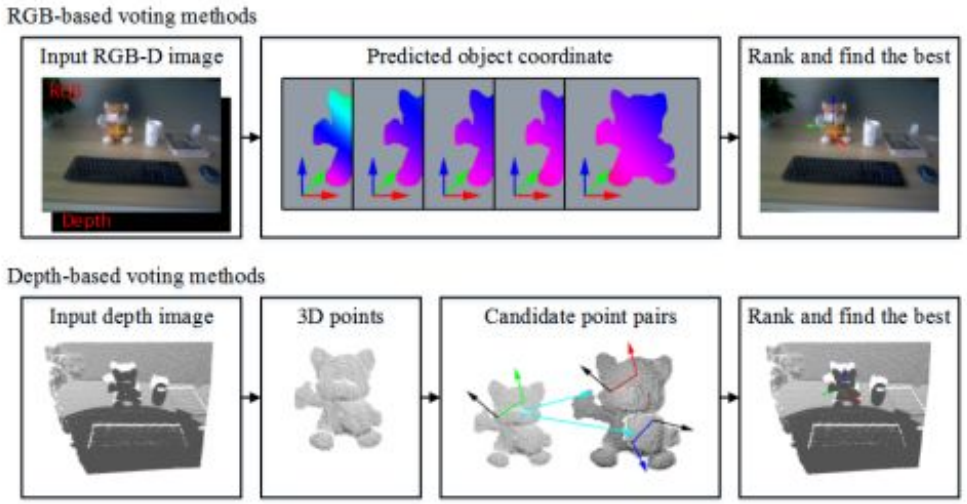
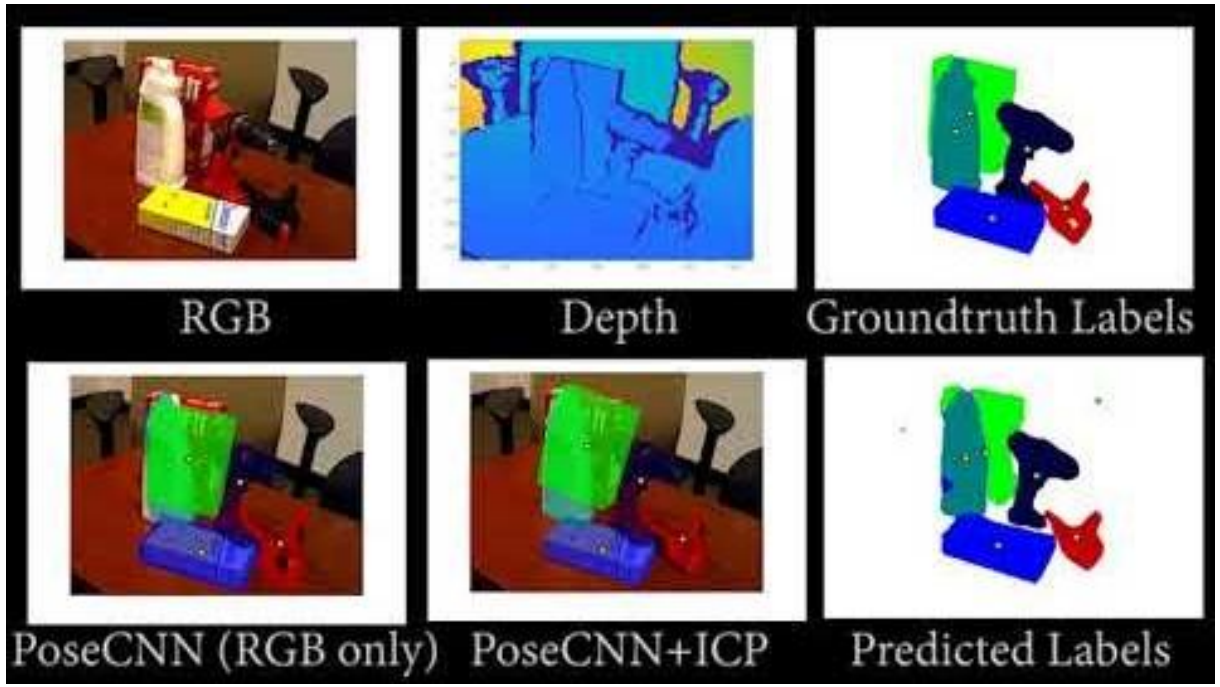


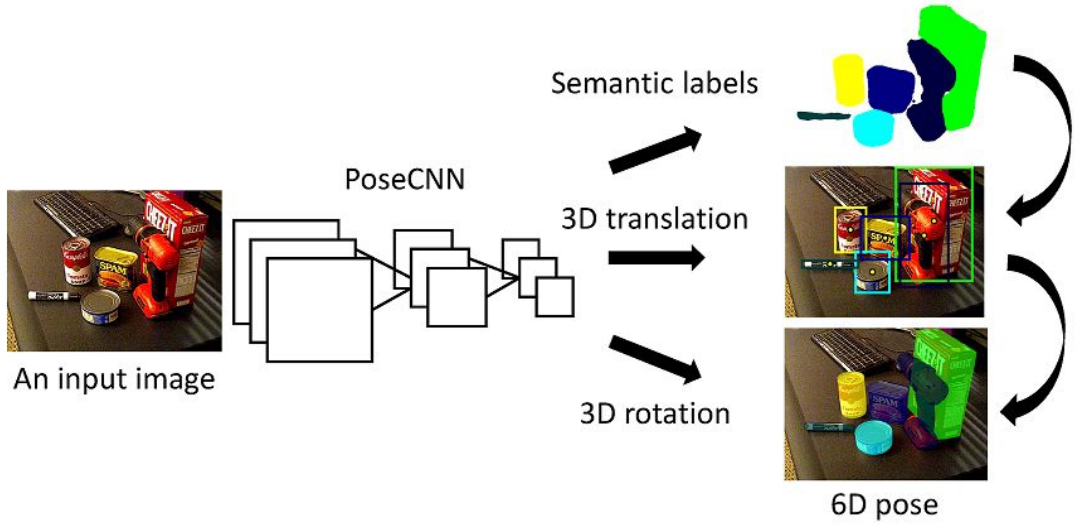
Figure 17: Typical functional flow-chart of voting-based object pose estimation methods without object detection.

Arxiv:1905.06658

Regression-based method

PoseCNN
- Localizing the center of objects
- Predicting its distance from camera
- Regressing rotation to a quarternion





Regression-based method

PoseCNN architecture

- Feature extraction
- Embedding
- Classification and regression

Two loss function for quaternion

- PoseLoss
- ShapeMatch-Loss

$$\mathrm{PLoss}(\tilde{\mathbf{q}}, \mathbf{q}) = \frac{1}{2m} \sum_{\mathbf{x} \in \mathcal{M}} \|R(\tilde{\mathbf{q}})\mathbf{x} - R(\mathbf{q})\mathbf{x}\|^2,$$

$$\mathrm{SLoss}(\tilde{\mathbf{q}}, \mathbf{q}) = \frac{1}{2m} \sum_{\mathbf{x}_1 \in \mathcal{M}} \min_{\mathbf{x}_2 \in \mathcal{M}} \|R(\tilde{\mathbf{q}})\mathbf{x}_1 - R(\mathbf{q})\mathbf{x}_2\|^2.$$

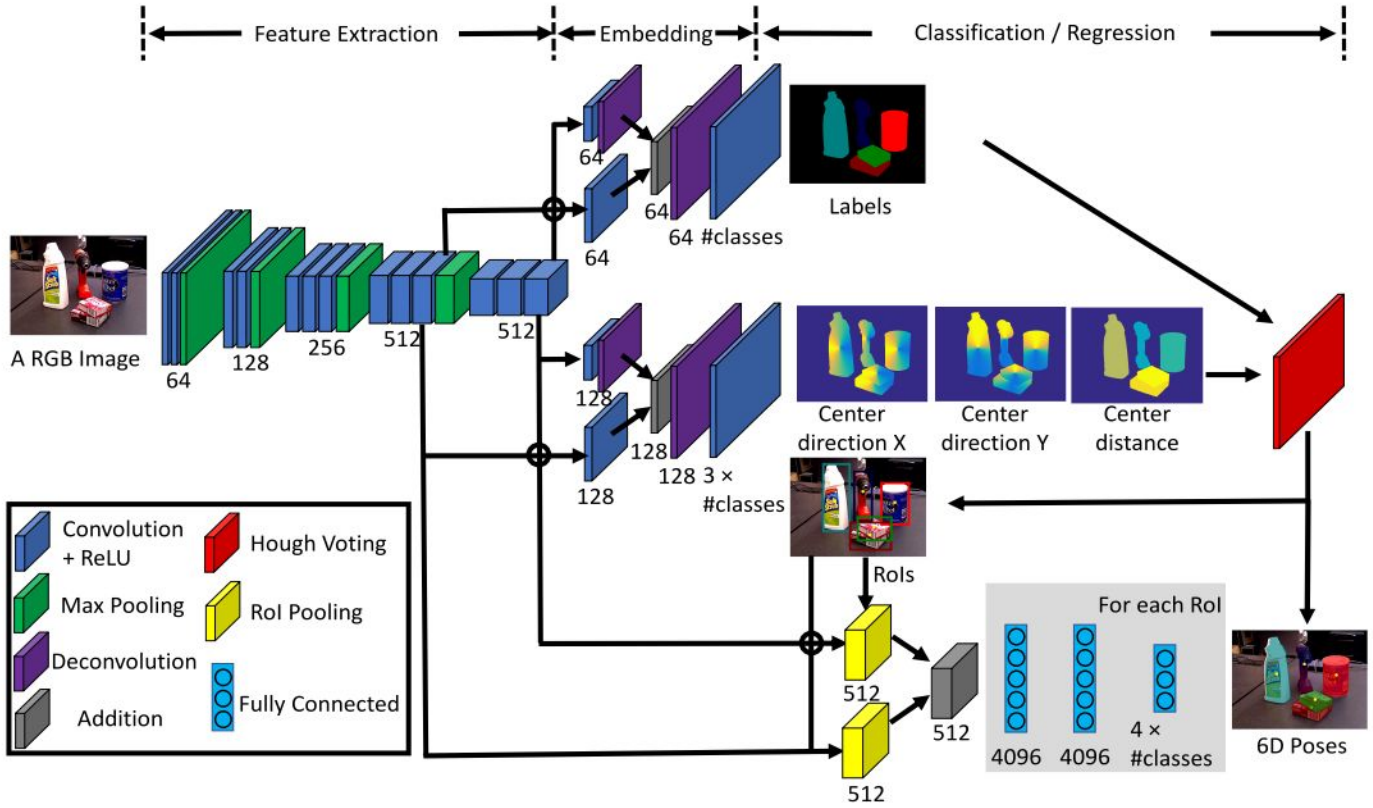


Fig. 2. Architecture of PoseCNN for 6D object pose estimation.

Regression-based method

DeepIM
Iterative refinement using DNN

Enable to combine other estimator

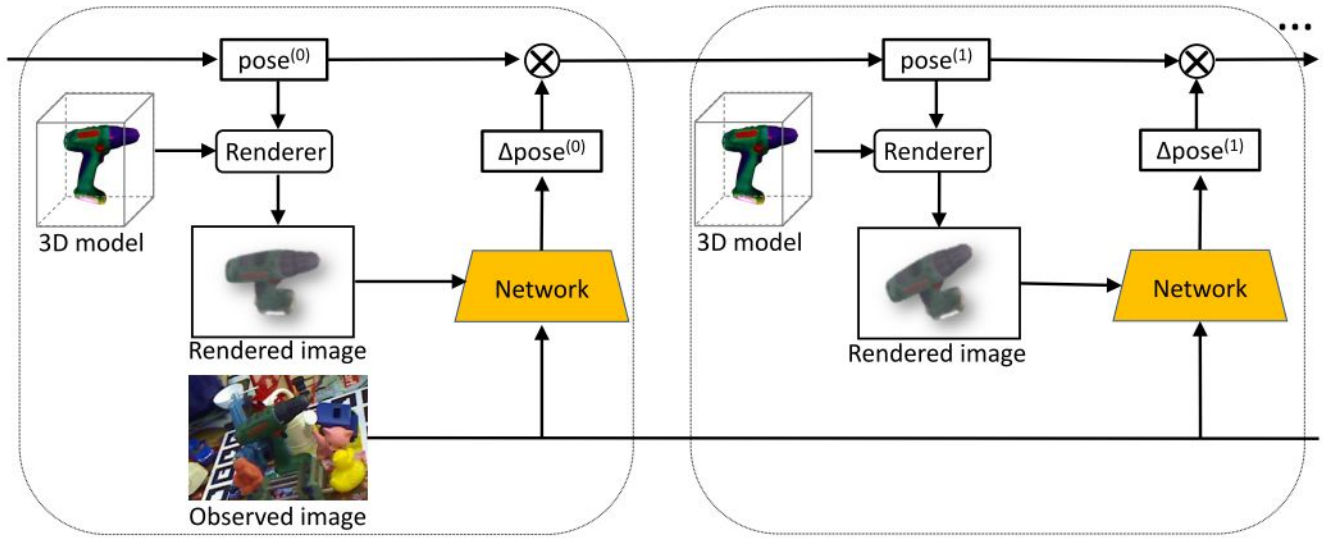


Fig. 1: We propose DeepIM, a deep iterative matching network for 6D object pose estimation. The network is trained to predict a relative SE(3) transformation that can be applied to an initial pose estimation for iterative pose refinement. Given a 6D pose estimation of an object, which can be the output of other pose estimation methods like PoseCNN (Xiang et al., 2018) ($\text{pose}^{(0)}$ in the figure) or the refined pose from previous iteration ($\text{pose}^{(1)}$ in the figure), along with the 3D model of the object, we generate the rendered image showing the appearance of the target object under this rough pose estimation. With the image pairs of rendered image and observed image, the network predicts a relative transformation (Δpose in the figure) which can be applied to refine the input pose. The refined pose can be used as the input pose of next iteration and therefore the process can be repeated until the refined pose converges or the number of iterations reaches a pre-determined number.

Regression-based method

DeepIM
Iterative refinement using DNN

Flownet: estimate optical flow

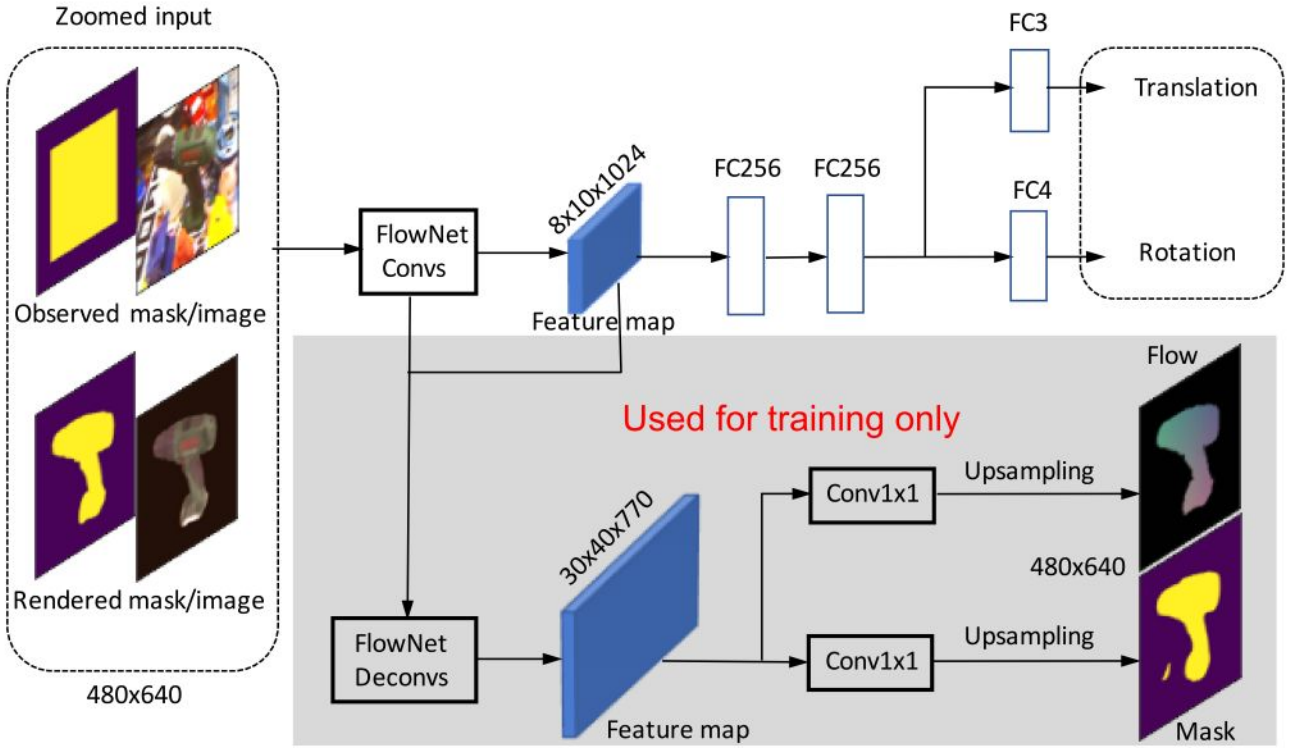
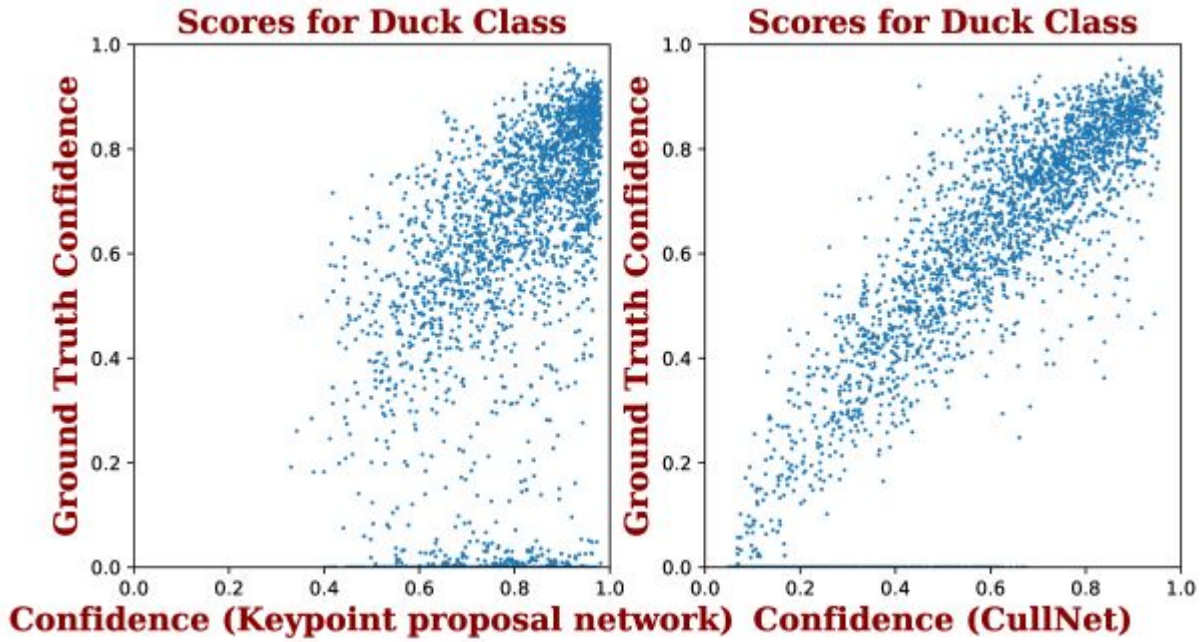


Fig. 3: DeepIM uses a FlowNetSimple backbone to predict a relative SE(3) transformation to match the observed and rendered image of an object. Taking observed image and rendered image and their corresponding masks as input, the convolution layers output a feature map which then be forwarded through several fully connected layers to predict the translation and rotation. The same feature map, combined with feature maps in the previous layers, will also be used to predict flow and foreground mask during training.

Regression-based method

CullNet

Single view image-based object pose estimation
Culling false positives
Calibrate the confidence score



Correspondence-based and Regression-based method

CullNet Architecture

Using Yolov3 for keypoints

Cullnet output confidence of objects pose by PnP.

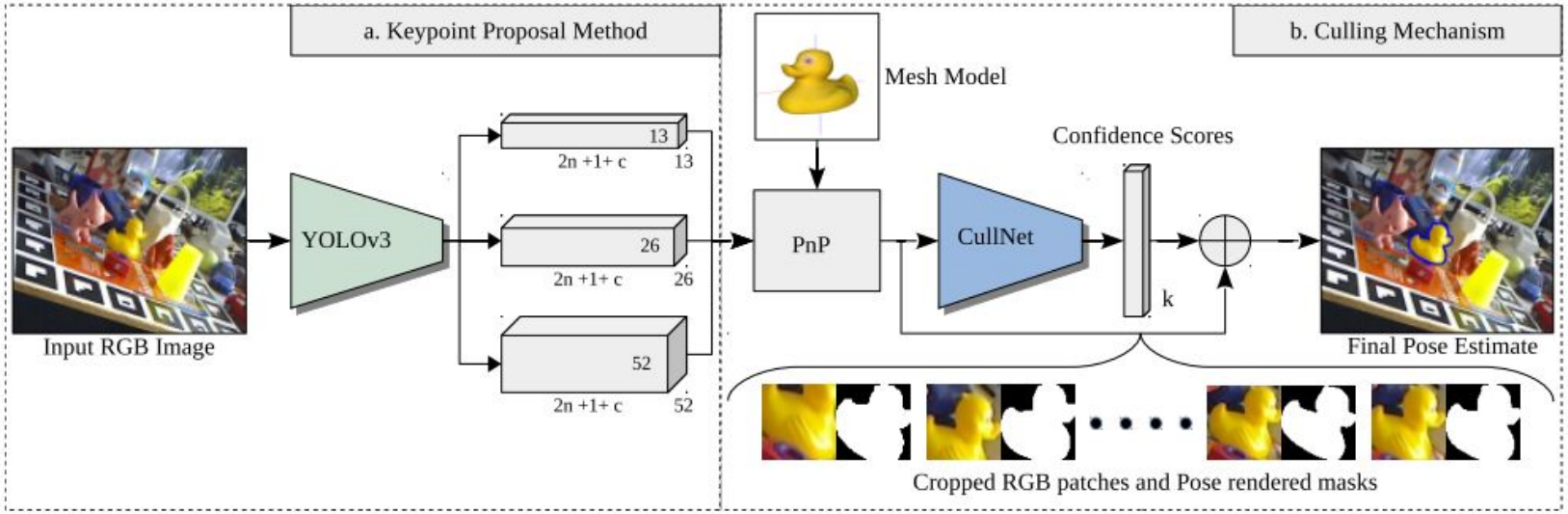
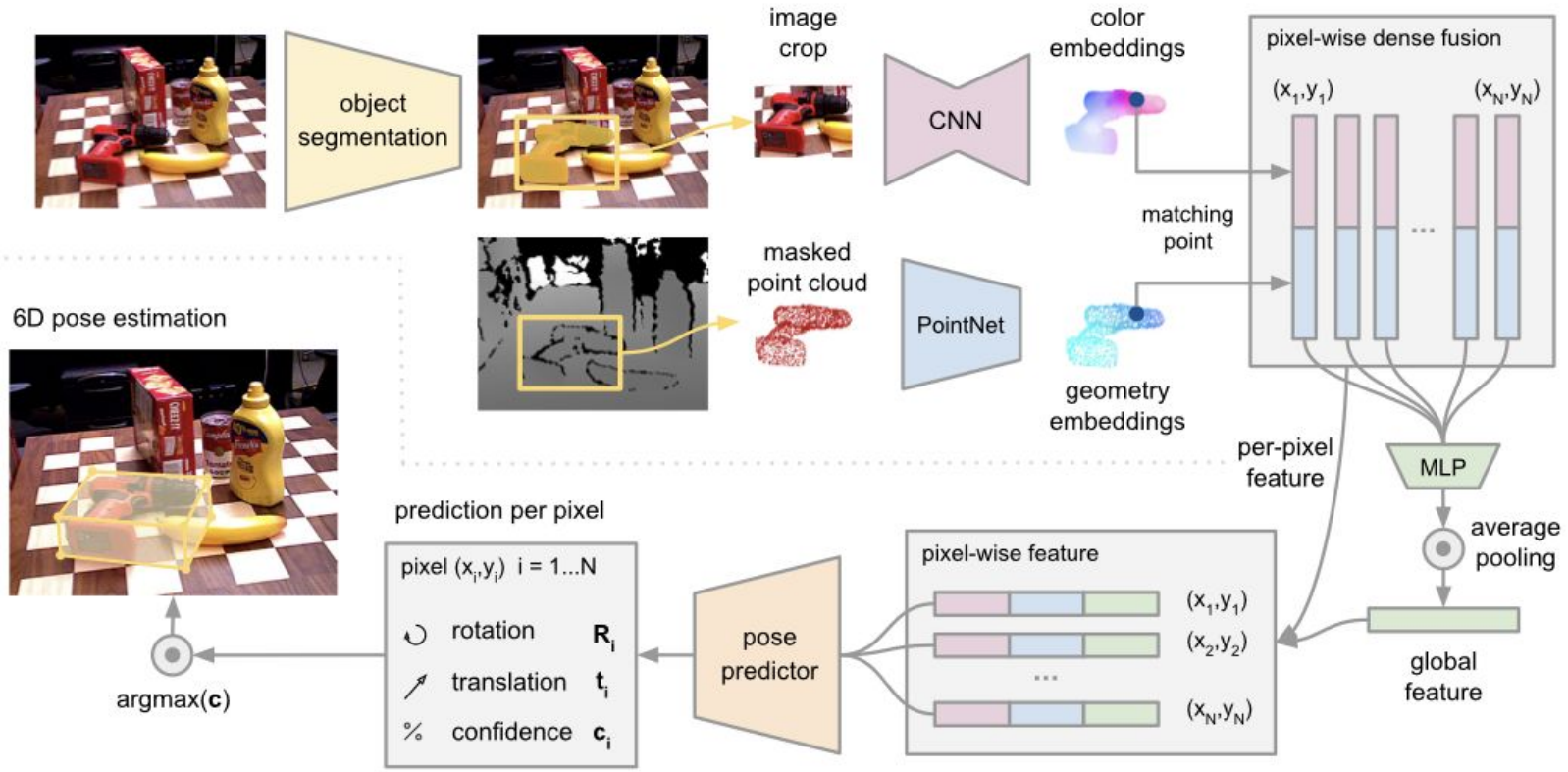


Figure 2: Overview of our pose estimation pipeline. Our approach operates in two stages: a) three 3D tensors are outputted using a YOLOv3 based architecture at 3 different scales in the form $2n + 1 + c$ outputs along a spatial grid of each tensor. b) using k sets of 2D keypoint proposals, k pose proposals are estimated using the E-PnP algorithm, then the original image and the pose rendered mask are cropped tightly fitting the rendered mask. Cropped RGB patches concatenated with the corresponding pose rendered masks are passed to CullNet to output calibrated object confidences. Calibrated confidences are finally used to pick the most confident pose estimate.

Voting-based and Regression-based method

DenseFusion
Estimating 6D pose of known objects



Comparison
ADD (Average Distance of Model Point)
ADD-S (for symmetric objects

$$e_{ADD} = \underset{x \in M}{avg} \left\| (Rx + T) - (\hat{R}x + \hat{T}) \right\|$$

Error smaller than threshold(< 2cm)

From the tables, we can see that DenseFusion achieved the highest accuracy comparing with other regression-based methods.
novel local feature fusion scheme using both RGB and depth images

Table 12: Accuracies of regression-based methods using ADD metric on LineMOD dataset.

Category	Method	Average
RGB-based methods	BB8 [Rad and Lepetit, 2017]	62.7
	SSD-6D [Kehl *et al.*, 2017]	79
	Tekin et al. [Tekin *et al.*, 2018]	55.95
	PoseCNN [Xiang *et al.*, 2017]	62.7
	PoseCNN+DeepIM [Xiang *et al.*, 2017; Li *et al.*, 2018c]	88.6
RGB-D-based methods	Implicit [Sundermeyer *et al.*, 2018]+ICP	64.7
	SSD-6D [Kehl *et al.*, 2017]+ICP	79
	PointFusion [Xu *et al.*, 2018]	73.7
	DenseFusion [Wang *et al.*, 2019a](per-pixel)	86.2
	DenseFusion [Wang *et al.*, 2019a](iterative)	94.3

Arithmer 株式会社
〒106-6040
東京都港区六本木一丁目６番１号 泉ガーデンタワー 38/40F(受付)
03-5579-6683
https://arithmer.co.jp/

Arithmer