Space Rendezvous Laboratory
Department of Aeronautics and Astronautics
Stanford University
https://slab.stanford.edu/

ESA Pose Estimation Challenge 2019

Tae Ha "Jeff" Park, Simone D'Amico

Jul. 03, 2019

Technical Report No: TN-19-01

Contents

Scope

This technical report summarizes and discusses the overall architecture developed for the best submission to the ESA Pose Estimation Challenge 2019[1]. The report ends with the brief discussion of the preliminary results. The proposed architecture has scored 4^{th} place for both synthetic and real test sets.

[1] https://kelvins.esa.int/satellite-pose-estimation-challenge/

1 Overall Architecture

The overall architecture is visualized in Fig. 1 in four steps. The core of the architecture is the regression of the 2D locations of the surface keypoints which can be used in conjunction with corresponding 3D model coordinates to solve the Perspective-n-Point (PnP) problem to extract full 6D pose estimates. The rest of the report provides detailed explanations of each steps.

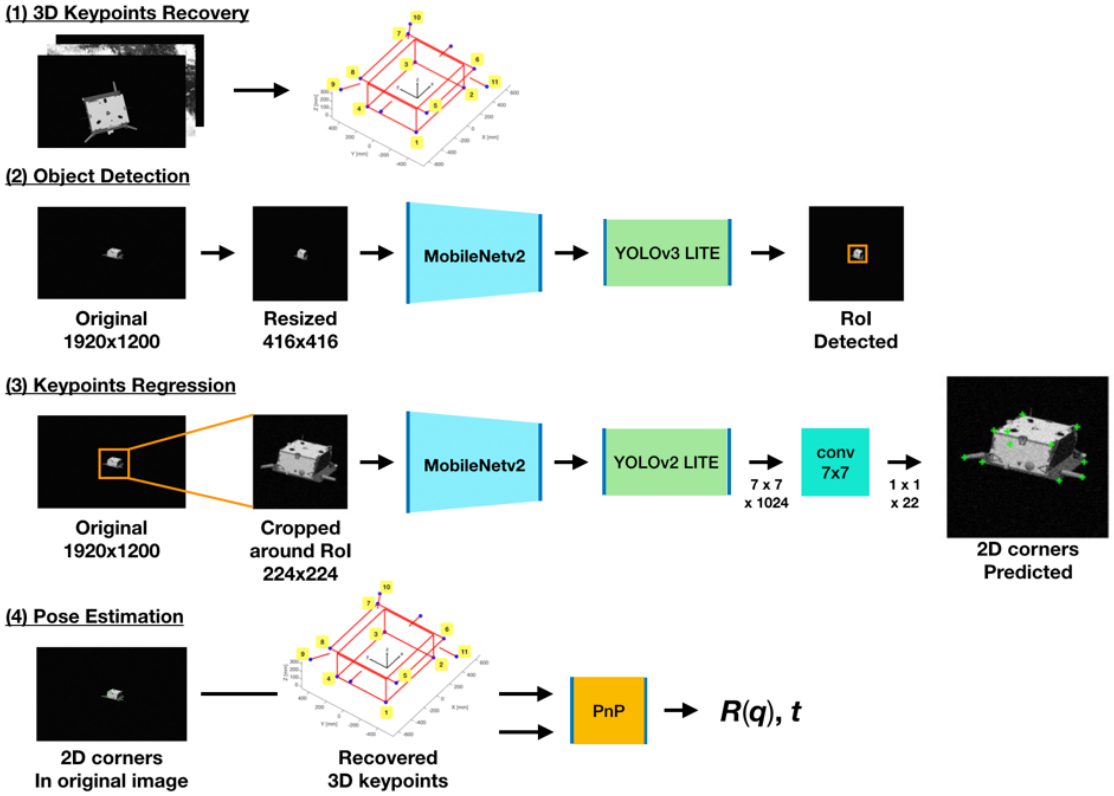

Figure 1: Visualization of the proposed architecture.

1.1 3D Keypoints Recovery

The competition does not provide the 3D model of the target, Tango spacecraft from PRISMA mission [1]. Fortunately, since the competition already provides 12,000 images of Tango with ground-truth pose labels, we can reconstruct the 3D model without difficulty. In our approach, since the method solves PnP with a fixed set of keypoints with known 2D-3D correspondences, it only suffices to recover the 3D model coordinates of those keypoints.

The keypoints selected for this project physically correspond to four corners of the bottom plate, four corners of the solar panel, and three ends of the antennae. The 11 keypoints are visualized in the wireframe model in Fig. 2. The choice of these keypoints is inspired by Zhao et al. [2] who claim the surface keypoints are presumably better than virtual keypoints, such as the corners of 3D bounding box, since they are more closely related to the target features.

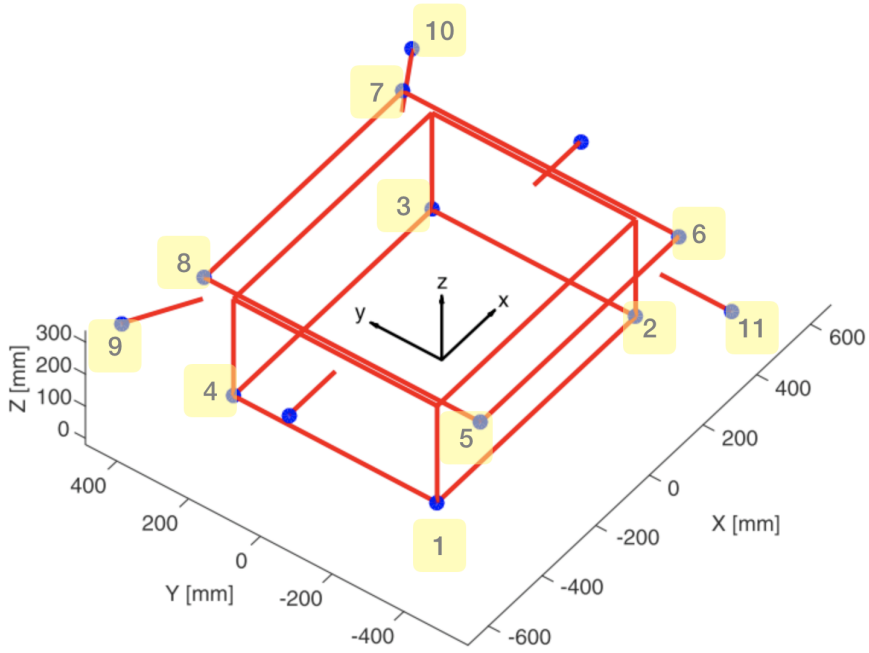

Figure 2: 11 Keypoints labeled on Tango wireframe model visualized in [3].

The method of recovery is rather brute-force. In order to recover the 3D model coordinates ($\boldsymbol{p}_{3D}$) of the aforementioned 11 keypoints, we first selected a set of 12 training images in which Tango spacecraft is well-lit and has varying poses. Then, we carefully and manually picked a set of visible keypoints ($\boldsymbol{p}_{2D}$). Then, the following optimization problem is solved to recover $\boldsymbol{p}_{3D}$:

$$\text{minimize} \sum_{j} ||s_j \boldsymbol{p}^h_{\text{2D},k} - \boldsymbol{K}[\boldsymbol{R}_j|\boldsymbol{t}_j]\boldsymbol{p}^h_{\text{3D},k}|| \tag{1}$$

i.e. for each k$^{\text{th}}$ point, minimize the sum of reprojection error over a set of images in which the k$^{\text{th}}$ point is visible. In Eq. 1, a superscript h indicates the point is expressed in homogenous coordinates, $\boldsymbol{K}$ is a known camera intrinsic matrix, $(\boldsymbol{R}_j, \boldsymbol{t}_j)$ is a known pose associated with j$^{\text{th}}$ image, and s_j is an optimization variable representing the scalar factor. Figure 3 shows the reprojection of the recovered keypoints onto the image plane. We can confirm the recovered 3D model points are quite accurate despite the manual selection of 2D keypoints.

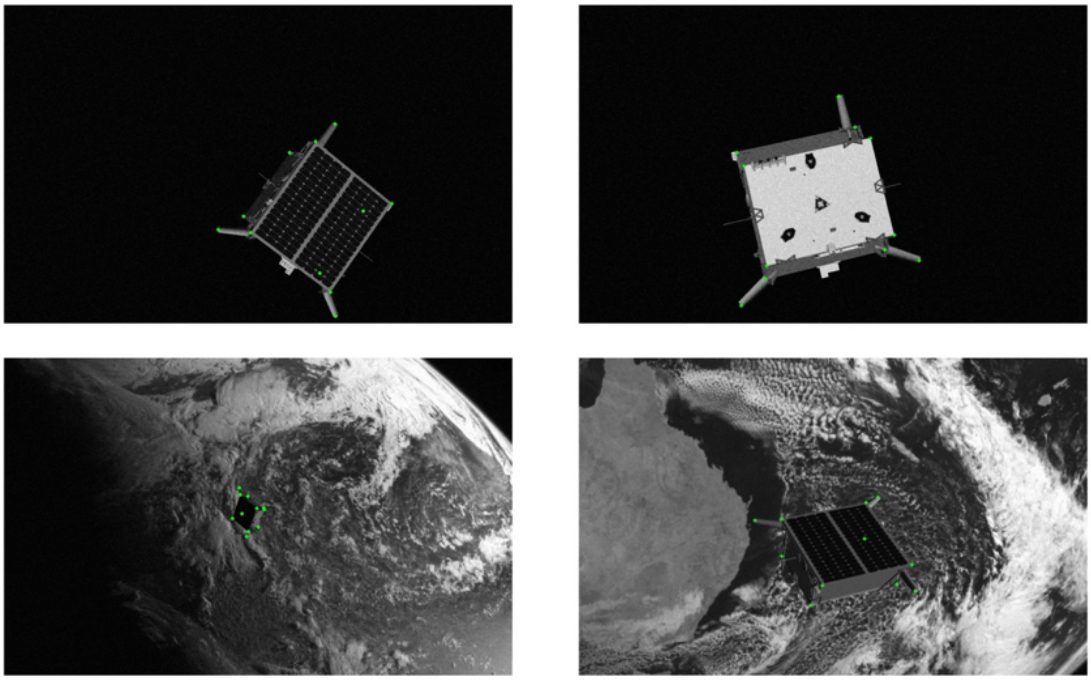

Figure 3: Examples of the recovered points reprojected on the images

1.2 Object Detection

The object detection network is used to detect the Region-of-Interest (RoI) and crop the original image around the target. The motivation of cropping before feeding to the keypoint regression network is that the SPEED images have the size of 1920x1200 while many conventional CNNs require fixed size input of 224x224, 416x416, etc. For such large images, the conventional way of downsizing the input image inevitably blurs much of the fine details of the target features which can help make more accurate predictions. This effect is especially strong when the object is farther away and

appears very small in the image. Thus, cropping from the original image ensures all those detailed features are retained when keypoint regression is performed in the next step.

The pipeline of the object detection network closely follows the architecture of state-of-the-art detection network YOLOv3 [4]. Since the SPEED images have a single known target, our YOLO-based network does not perform object classification. Instead, our network takes an image resized to 416x416 and outputs 13x13x5, 26x26x5, and 52x52x5 tensors at three different stages. Later two outputs are the results of upsampling from the previous stages and are designed to detect much smaller objects. The output tensors basically divide the image into 13x13, 26x26, and 52x52 grids, where each grid predicts $(t_0, t_x, t_y, t_w, t_h)$ that are related to the confidence score, $p(c)$, and four parameters of 2D bounding box, i.e. location (x, y) and dimension (w, h), via the following equations,

$$\begin{aligned} p(c) &= \sigma(t_0) \\ x &= \sigma(t_x) + g_x \\ y &= \sigma(t_y) + g_y \\ w &= p_w e^{t_w} \\ h &= p_h e^{t_h} \end{aligned}$$

where σ is a sigmoid function, (g_x, g_y) is the grid location, and (p_w, p_h) is the dimension of each anchor box. We pre-define nine anchors boxes, three for each prediction stage. Their sizes are determined using k-means clustering on the training set. The ground-truth confidence is set to 1 if the grid contains the object center and to 0 if otherwise. The readers are encouraged to refer to a series of publications on YOLO architecture and training details [5, 6, 4].

However, in this submission, couple modifications are made from the original YOLOv3.

- The backbone of the original YOLOv3 is Darknet-53, which consists of 53 3x3 and 1x1 convolutional layers. In this submission, MobileNetv2 [7] is used instead as a backbone. MobileNet is an innovative network structure consisting of depth-wise separable convolution operations [8] and inverted residual bottleneck blocks to drastically reduce the number of parameters. It shows real-time capability on smartphone CPUs with marginal loss of performance.
- The extra convolutional layers are also replaced with depth-wise convolutional layers (hence the name YOLOv3 LITE in Fig. 1).
- Since SPEED images guarantee the presence of a single object in all its images, there is no need to select detected objects using the confidence threshold or perform non-max suppression during the inference. The network simply selects a grid with the highest confidence to provide the correct bounding box location.
- The original YOLOv3 loss, excluding the loss for classification since we are

dealing with a single object detection, is

$$\lambda_{\text{coord}}\mathcal{L}_{\text{coord}} + \lambda_{\text{conf}}\mathcal{L}_{\text{conf}} \tag{2}$$

where $\mathcal{L}_{\text{coord}}$ is a sum of mean-squared errors (MSE) of bounding box parameters (x, y, w, h), and $\mathcal{L}_{\text{conf}}$ is a sum of binary cross-entropy error (BCE) between predicted and ground-truth confidence scores. Since the goal of object detection is to maximize the Intersection over Union (IoU) metric, it makes sense to use it as a loss function to minimize. Specifically, $\mathcal{L}_{\text{coord}}$ is replaced with the Generalized Intersection over Union loss ($\mathcal{L}_{\text{GIoU}}$) [9] which is defined as

$$\mathcal{L}_{\text{GIoU}} = 1 - GIoU, \text{ where } GIoU = \frac{I}{U} - \frac{A^C - U}{A^C}, \tag{3}$$

where I and U each represent the intersection and union areas, and A^C is the area of the smallest box enclosing both predicted and ground-truth bounding box. This loss formulation ensures even when there is no overlap (i.e. IoU = 0), the gradient is bigger as the separation between bounding boxes becomes larger.

Overall, these modifications allow consistently accurate bounding box regression with much fewer parameters thanks to the MobileNet architecture and depth-wise convolution operations. Moreover, minimizing GIoU score ensures even the worst prediction has nonzero overlap with the target, mitigating the effect of some of the largest prediction errors.

1.3 Keypoint Regression and Pose Estimation

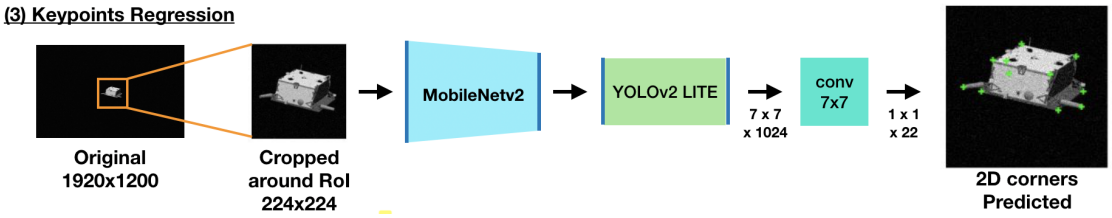

Figure 4: Step 3 from Fig. 1

The architecture of the keypoint regression network is very similar to the object detection network – it is kept light using MobileNetv2 and depth-wise convolution operation. However, it uses the lite version of YOLOv2 [6] instead of YOLOv3. The difference is that YOLOv3 makes predictions at three different scales in order to better detect the smaller objects, whereas YOLOv2 makes the prediction only once at the very end of the network. Predictiosn at multiple stages are not necessary since the input image, which is cropped around the object from the original image, should have the target centered and appearing large.

The input to the keypoint regression network is prepared by cropping the detected RoI from the original 1920x1200 image. Cropping from the original image helps maintain the fine features of the target, and we found that these fine features help regress keypoint locations more accurately. The cropped image is resized to 224x224 and fed into the YOLOv2-based keypoint regression network which outputs 7x7x1024 tensor. Here, a single 7x7 convolution is used to produce 1x1x22 tensor, where 22 corresponds to the 2D locations of 11 keypoints of the spacecraft recovered (See Fig. 2. I found the dimension reduction at the end using convolution performs better than using global average pooling. The loss function is simply a sum of MSE between predicted and ground-truth keypoint locations. The ground-truth locations are prepared by projecting the recovered 3D model points using the available pose labels. Then, during inference these predicted 2D keypoints are used in PnP with recovered 3D keypoints to extract the full 6D pose using EPnP algorithm [10].

2 Training Details

For submission, both networks are trained using the RMSprop optimizer with batch size of 48 and momentum and weight decay set to 0.9 and 0.00005, respectively. The learning rate is initially set to 0.001 and decay exponentially by a factor of 0.98 every epoch. All 12,000 synthetic training images are used to train the networks for submission; *no real images are used for training* in order to gauge the network's ability to generalize to dataset of different domain. The networks are trained on NVIDIA GeForce RTX 2080 Ti 12GB GPU.

For training both networks, data augmentation is implemented by randomly changing the brightness and contrast and applying gaussian noise. For keypoint regression, the size of ground-truth RoI is enlarged randomly up to a factor of 50% then randomly shifted. These augmentations have the effect of varying the resolution of the spacecraft and making the network robust to object translation and misaligned RoI detection.

3 Results

The overall architecture scored **4**$^{\text{th}}$ place in the competition for both real and synthetic test sets. The final SPEED scores on entire test sets are **0.0626** for synthetic and **0.3951** for real test set. Table 3 summarizes the specs of object detection and keypoint regression networks.

We can see that compared to a deep network like YOLOv2, which has a reported number of parameters of about 50M, our combined networks have much fewer parameters thanks to the MobileNet architecture. Even though both object detection and keypoint regression networks both use MobileNetv2 as a backbone network, we have not considered sharing the weights for both networks because they are exposed to fundamentally different resolutions of the target features.

Altogether, our method is fast - about 70 FPS on a GPU and 4 FPS on an Intel(R) Core(TM) i9-9900K CPU at 3.60 GHz for inference. A sharp jump in CPU speed for detection network is likely due to the upsampling operations. A deep yet light CNN architecture like MobileNet can potentially pave the way towards the implementation of deep CNN-based algorithms on-board the spacecraft with limited computing resources.

Network	Number of parameters	Speed (ms) on GPU	Speed (ms) on CPU
Object Detection	5.53M	7	230
Keypoint Regression	5.64M	7	32

Table 1: Results of the proposed method. Inference speed only accounts for the forward propagation of detection and keypoint regression networks withot any post-processing steps.

4 Disclaimer

The method proposed in this report is developed and tested only using the SPEED dataset [3] provided under the license[2].

[2]https://creativecommons.org/licenses/by-nc-sa/3.0/

5 References

[1] Simone D'Amico, Per Bodin, Michel Delpech, and Ron Noteborn, "Prisma", in *Distributed Space Missions for Earth System Monitoring Space Technology Library*, Marco D'Errico, Ed., vol. 31, chapter 21, pp. 599–637. 2013.

[2] Zelin Zhao, Gao Peng, Haoyu Wang, Haoshu Fang, Chengkun Li, and Cewu Lu, "Estimating 6d pose from localizing designated surface keypoints", *ArXiv*, vol. abs/1812.01387, 2018.

[3] Sumant Sharma and Simone D'Amico, "Pose estimation for non-cooperative rendezvous using neural networks", in *2019 AAS/AIAA Astrodynamics Specialist Conference, Ka'anapali, Maui, HI*, January 13-17 2019.

[4] Joseph Redmon and Ali Farhadi, "Yolov3: An incremental improvement", *CoRR*, vol. abs/1804.02767, 2018.

[5] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[6] Joseph Redmon and Ali Farhadi, "Yolo9000: Better, faster, stronger", *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[7] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks", in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 4510–4520.

[8] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications", 2017, cite arxiv:1704.04861.

[9] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese, "Generalized intersection over union", June 2019.

[10] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua, "EPnP: An Accurate O(n) Solution to the PnP Problem", *International Journal of Computer Vision*, vol. 81, no. 2, pp. 155–166, 2008.