Probabilistic Object Tracking using Depth Camera

Jan Emrich [*1] **Simon Kiefhaber** [*1]

Abstract

In this work, we address the problem of 6DoF pose tracking of multiple objects using images produced by a depth camera. This is achieved by utilizing a hierarchical approach that consists of a low-level detector that detects objects in depth images and a high-level tracker that tracks multiple objects over time. The low-level detector utilizes a YOLOv5 object detector and uses its predictions and combines them with a particle filter and a Bayesian network to predict 6DoF poses for all objects in the scene. The high-level tracker processes these poses with a probability hypothesis density(PHD) filter, which allows us to track an arbitrary number of objects in the scene.

1. Introduction

Many robotic manipulation tasks require a recognition and an estimation of the poses of multiple objects. Each pose of an rigid object consists of a position and a orientation of an object in space. Both, the position and orientation have three degrees of freedom(DoF) and therefore this pose is also called a 6DoF pose.

Tracking these poses over time allows to add a feedback to the controllers that are used in applications like robotic assembly. This should make these robots more robust to external disturbances than a robot with an open loop controller would be. Further, a task like this requires the detection of new objects in the scene and it requires the handling of occluded objects because many objects are moved and when assembling they occlude parts of each other.

Probabilistic approaches are able to track arbitrary objects given a 3D model with high precision through filtering of depth images. They also incorporate methods to deal with occlusion, but they cannot initialize in real-time without a strong prior and therefore they are not very efficient in detecting new objects. In a robotic assembly scenario the number of objects varies, which gives rise to the need for a tracking system that can handle multiple objects.

Therefore, we investigate an hierarchical filter with a probability hypothesis density (PHD) filter as the upper level. PHD filters have a united probability potential for all objects with the property that the integral of the potential equals the number of estimated objects in the scene. The PHD filter tracks on top of the lower level tracker, for which we use a particle filter and a Bayesian network to find the 6DoF poses of objects.

As these trackers cannot initialize arbitrarily in this high-dimensional problem, we detect objects using a deep learning detector on the RGB images. The detector predicts bounding boxes. On these bounding boxes we iteratively estimate poses and introduce them to the PHD filter.

2. Related Work

There are different approaches to 6DoF object tracking in the literature. Approaches like the PoseCNN (Xiang et al., 2018) are able to track multiple objects with a high accuracy from RGB-D images, incorporating convolutional neural networks and end-to-end training. The downside of these approaches is that they are limited to tracking objects they were trained on and therefore this approach is not able to track arbitrary objects given a 3D model.

It has been demonstrated that it is possible to use probilistical graphical models for the tracking of arbitrary objects based on depth images (Wuthrich et al., 2013; Issac et al., 2016).

A dynamic Bayesian network, that models camera noise and occlusions explicitly, can be utilized in a combination of approximate inference, particle filters and Bayesian filters to track a single moving object in a scene (Wuthrich et al., 2013). Another approach is to directly apply a Gaussian filter on the depth images (Issac et al., 2016). To do this, a stochastic model for the object movement and an observation model, which models the probability of a pose given an image, is defined. A ”heavy-tailed gaussian”, which is a sum of a uniform distribution in the working range of the camera and a gaussian distribution, is used to model the camera noise and occlusions in a single step, but the

[*]Equal contribution [1]TU Darmstadt. Correspondence to: Jan Emrich <jan.emrich@stud.tu-darmstadt.de>, Simon Kiefhaber <simon.kiefhaber@stud.tu-darmstadt.de>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

covariance of this distribution is very large, which leads to standard Gaussian filters failing. This is solved by using robust gaussian filters (Wuthrich et al., 2016).

Both of the previously mentionend methods are limited to track single objects in real-time because they are based on single-object Bayes filters and their computationally complexitiy is too high to handle many different objects.

The probability hypothesis density(PHD) was derived as a generalization of the Bayes filter for tracking an arbitrary number of objects with multiple sensors (Mahler, 2003), but it is challenging to evaluate the probability hypothesis density function, because it uses multiple integrals with no known closed-form solution and it is necessary to find local optima in this function to get a prediction of the location of an object.

The Sequential Monte Carlo PHD(SMC-PHD) filter (Vo et al., 2005) is a particle filter-based implementation of the PHD filter which makes it possible to use the PHD filter by just evaluating single points in the PHD function. This approach was further developed into the Box-Particle PHD filter (Schikora et al., 2012) where interval analysis is used to evaluate volumes instead of point-masses for each particle. The advantage of this approach is that it needs less particles than the SMC-PHD filter, while achieving better results.

Another approach to make the PHD filter tractable is the GM-PHD filter (Vo & Ma, 2006) where the PHD function is modeled by a gaussian mixture model. This approach was further developed in (Clark et al., 2006) to gain better performances in cluttered scenes.

A problem of all of these approaches to make the PHD filter tractable is that it was just demonstrated that they do work for simplified tracking tasks in two dimensional space.

It was also demonstrated that it is possible to apply the PHD filter on more complex and realistic tasks, like pedestrian and vehicle tracking (Edman et al., 2013; García et al., 2018) and the usage of a GM-PHD filter in a hierarchical approach was demonstrated (Song & Jeon, 2016), but to our knowledge neither the GM-PHD filter nor the SMC-PHD filter have been applied to multi-target 6DoF pose tracking.

Since it was shown that the SMC-PHD filter works very well on simplified simulations, we decided on using these combined with Bayesian networks for the tracking of multiple objects in this work.

3. The PHD Filter

This section describes the basics of the SMC-PHD filter, which is used by our tracking approach.

The intuition behind the PHD filter is that it embeds all of the measurements it gets from multiple sensors into one function that maps a location in the state-space onto a (unnormalized) probability of a tracked object being present at that location. Further, the sum over an area of the function is equal to the expected number of objects in that region. A function like this allows to track objects by tracking the peaks of the function.

As discussed there are variants of the PHD filter which make assumptions to make it tractable. In algorithm 1 we show one iteration of the SMC-PHD filter (Vo et al., 2005) for completeness. Note that $w_{j,i}$ describes a temporary variable while w_i is the weight of a particle, to keep the notation of (Schikora et al., 2012).

The following equations are used in the algorithm for timestep k, measurements z_j and estimated target states $\widetilde{x_i}$. N_k is the total number of particles at time step k. Likelihoods of particle states are normalized with this correction term

$$\lambda_{k|k-1,j}(z_j) = \sum_{i=1}^{N_k} p_k(z_j|\tilde{x_i}) p_k^D(\tilde{x_i}) w_i, \tag{1}$$

where p_k is the likelihood of a target z and p_k^D is the probability of detection for a given type of target.

Weights per measurement j and particle i are computed

$$w_{j,i} = \frac{p_k(z_j|\tilde{x_i})}{\lambda_{k|k-1,j}(z_j)} \cdot w_i \tag{2}$$

to obtain the marginal probability of a target j

$$W_j = \sum_{i=1}^{N_k} w_{j,i} \tag{3}$$

and estimate target states

$$y_j = \sum_{i=1}^{N_k} \tilde{x_{j,i}} \cdot w_{j,i}. \tag{4}$$

Covariance matrices are computed as follows

$$C_j = \sum_{i=1}^{N_k} w_{j,i}[(\tilde{x_i} - \hat{y_j})(\tilde{x_i} - \hat{y_j})]. \tag{5}$$

The filter update equation is

$$w_i = [(1 - p_k^D(x_{j,i})) + \frac{p_k(z|x_{j,i}) p_k^D(x_i)}{\lambda_{k|k-1,j}(z)}] \cdot w_i. \tag{6}$$

Algorithm 1 SMC-PHD Filter

Parameters: Number of total particles N_k, Number of new particles $N_{k,new}$
Input: Set of new measurements Z, Set of particles P
Output: Set of target estimates Y, Set of particles P
for $i = 1$ **to** N_k **do**
  $\widetilde{x_i}$:= dynamicModel($\widetilde{x_i}$)
end for
$P := P \cup initParticles(C_J, Z, N_{k,new}))$
for $j = 1$ **to** $|Z|$ **do**
  $\lambda_{k|k-1,j} := correctionTerms(z_j, P, j)$ (Eq. (1))
end for
for $j = 1$ **to** $|Z|$ **do**
  for $i = 1$ **to** N_k **do**
    $w_{j,i} := weights(z_j, p_i, \lambda_{k|k-1,j})$ (Eq. (2))
  end for
  $W_j := marginalize(w_{j,i})$ (Eq. (3))
end for
for $j \in \{j|W_j > \tau\}$ **do**
  $y_j := estimateTargets(p_i, w_{j,i})$ (Eq. (4))
end for
$C_j := computeCovariance()$ (Eq. (5))
for $i = 1$ **to** N_k **do**
  $P := update(Z, x_i, \lambda_{k|k-1,j})$ (Eq. (6))
end for
$\eta := \sum_{i=1}^{N_k} w_i$
$P := resample(P)$
for $i = 1$ **to** N_k **do**
  $w_i := \frac{\eta}{N_k}$
end for

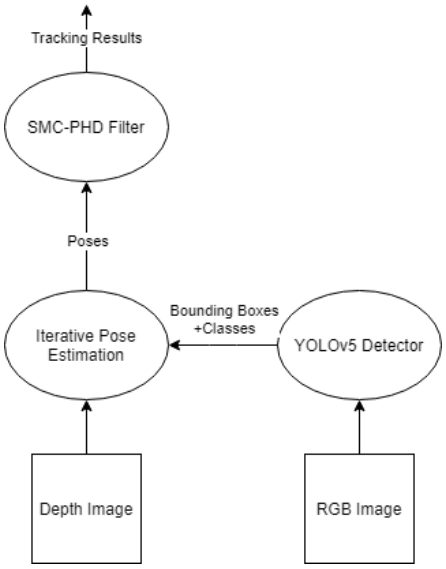


Figure 1. The architecture of our proposed method.

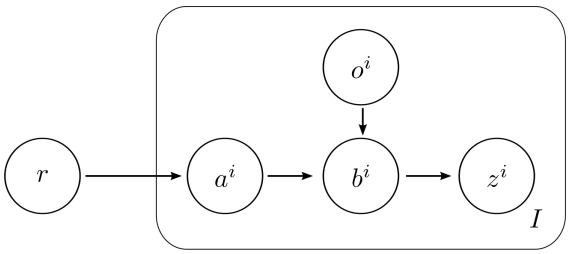


Figure 2. Visualization of the Bayesian network, which can be used to calculate the likelihood of an observation given a pose. **Source:** (Wuthrich et al., 2013)

4. Methods

In this section, we will describe the individual parts of our proposed tracking framework. Section 4.1 describes a modification of the approach proposed in (Wuthrich et al., 2013) to process images of a depth camera and predict the poses of each object in the image. These poses are then processed by a PHD filter based approach, described in section 4.2, to track objects across multiple images. This is also shown in figure 1.

4.1. Low-Level Object Detector

The goal of the low-level object detector is to predict the poses and object types of the objects in the image.
This is split into two stages, where the first stage runs an object detection algorithm on the RGB-image, produced by the depth camera, to find bounding boxes in the image containing an object and to find the type of the object inside of the bounding box. In this work, we utilize a YOLOv5 (Jocher et al., 2021) network, which is a neural network based and fast method for object detections.

The second stage uses the depth information of the previously identified bounding boxes to find the poses of the objects utilizing a probabilistic graphical model.

The graphical model we use is identical to the observation model used in (Wuthrich et al., 2013) and it is shown in figure 2.

The random varible r corresponds to an objects pose, z^i denotes the depth measurement at the point i, o^i is added to the graph to model whether the point i is occluded or visible. The variable b^i is the distance to the pixel i and a^i is the distance to the tracked object with the pose r applied to it.

The independency assumptions made in the Bayesian network allow the modeling of the noise that is present due to errors in the 3D model of the object $p(a^i|r)$ and $p(b^i|a^i, o^i)$, which defines the probability of an pixel showing the tracked object rather than an object that occludes it independently.

It also allows to integrate out the variable a^i and b^i by

$$p(z^i|r, o^i) = \int_{a,b} p(z^i|b^i)p(b^i|a^i, o^i)p(a^i|r)$$

which can be done prior to using the Bayesian network for the object tracking.

We use this network in an iterative approach to find the pose of the object in the depth image. Therefore, we crop out the region in the image that contains an object according to the YOLOv5 object detector. We then initialize multiple poses such that the x and y positions are set to the center of the bounding box and small disturbances sampled from $\mathcal{N}(0, \sigma_{x,y})$ are added to these positions. The z positions are set to the average depth of the cropped out section of the depth image and there we also add a disturbance sampled from $\mathcal{N}(0, \sigma_z)$. Since the information we receive from our object detector does not contain any information about the rotation of the object, we have to start with uniformly sampled rotations. By sampling $u_1, u_2, u_3 \sim U(0, 1)$, we can calculate

$$h = \begin{pmatrix} \sqrt{1-u_1}\sin(2\pi u_2) \\ \sqrt{1-u_1}\cos(2\pi u_2) \\ \sqrt{u_1}\sin(2\pi u_3) \\ \sqrt{u_1}\cos(2\pi u_3) \end{pmatrix}$$

which is a uniformly sampled and normalized quaternion (Shoemake, 1992).

Next, we evaluate all of the generated poses using the Bayesian network and do a priority resampling of the weights based on the likelihood computed by the Bayesian network for each proposed pose and then a small perturbance is added to the new pose candidates and the process is repeated again. This resembles a particle filter and a pseudocode of our low-level object detector can be found in algorithm 2.

4.2. High-Level Object Tracking

The purpose of the high-level object tracker is to track the objects in the scene between the frames. Therefore it receives the poses created by the low-level detector. These poses are then used as measurement in a SMC-PHD filter, where these measurements are than embedded into the PHD function. The peaks, which are tracked by the particles of the SMC-PHD filter, then correspond to the locations and orientations of the tracked objects.

Our SMC-PHD filter is the same algorithm as 1 except following changes to adapt to 6D poses and our specific use case.

The state space of the SMC-PHD filter is a 6D representation of the pose.

Algorithm 2 Low-Level Object Detector

Input: Depth image I, Number of particles P, Number of iterations N
boundingBoxes = YOLOv5.detect(I)
for each boundingBox ∈ boundingBoxes **do**
  region = I[boundingBox]
  for $i = 1$ **to** P **do**
    Initialize particles[i]
    particles[i].x = boundingBox.midX() + $\mathcal{N}(0, \sigma_{x,y})$
    particles[i].y = boundingBox.midY() + $\mathcal{N}(0, \sigma_{x,y})$
    particles[i].z = region.mean() + $\mathcal{N}(0, \sigma_z)$
    particles[i].orientation = uniformQuaternion()
  end for
  for iteration = 1 **to** N **do**
    likelihoods = evaluate(particles, region)
    particles = resample(particles, likelihoods)
    particles = perturb(particles)
  end for
  poses[i] = particles[likelihoods.argmax()]
end for
return poses

We treat the position and orientation parts of the state space differently in the *estimateTargets* function. Positions are averaged exactly like in 1, while orientations, which are represented as quaternions, are averaged according to (Markley et al., 2007).

The dynamic model and measurement model ($p_k(z|x_i)$) are both modeled as a multivariate gaussian with six dimensions for the six dimensions of the pose. We assume the dimensions are independent. The variance for each dimension is estimated beforehand from detection data given by the lower-level filter. The orientation variances are estimated based on the Euler angels representation. In both the dynamic and measurement model we assume small differences in angles and can therefore linearize and safely use the Euler angles representation.

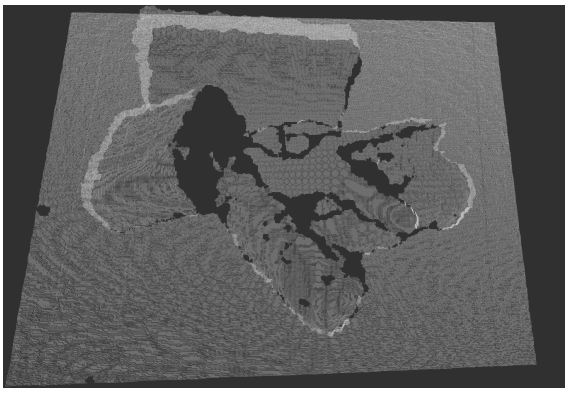
5. Experiments

5.1. Dataset

We evaluate our proposed approach on the "YCB-Video Dataset" (Xiang et al., 2018) which consists of $113,827$ RGB and depth images in 91 video sequences captured by an "Asus Xtion Pro Live RGB-D" camera with a resolution of 640×480 pixels at 30 frames per second.

The dataset uses 21 different objects, like different types of food cans, boxes and different tools. In each scene a subset of these objects is lying on a table, where they may

(a) RGB image



(b) Depth cloud

Figure 3. Visualization of one sample from the YCB-Video dataset.

also occlude each other, and the camera rotates around the objects in each sequence. One example RGB images from this dataset can be seen in 3.

Further, this dataset provides 6DoF poses for each object in each image and it defines $2,949$ key frames from 12 of these sequence that should be evaluated to make different approaches using this dataset comparable.

The YCB benchmark (Çalli et al., 2015b; 2017; 2015a) provides 3D scans of these objects, which we down-sample to a maximum of 860 vertices, because the original 3D models consist of an average of $269,808$ vertices which would significantly slow down our approach without yielding any advantage for our algorithm because there are far more vertices per object than there are pixels from the depth camera.

The YCB-Video dataset does not provide a noise model of the camera used, so we use the noise model from (Wuthrich et al., 2013) which is a noise model for the same camera type as used for the acquisition of the dataset.

5.2. Metrics

We use the ADD metric proposed by (Xiang et al., 2018) to evaluate the accuracy of the pose of a single object in the scene. It is defined by

$$\text{ADD} = \frac{1}{|M|} \sum_{x \in M} ||(Rx + T) - (\tilde{R}x + \tilde{T})||$$

where M is the set of vertices in the mesh, R is the ground truth rotation, T the ground truth translation, $\tilde{R}$ and $\tilde{T}$ are the predicted rotation and translation, respectively. This metric calculates the average distance between each point of the mesh, transformed by the ground truth, and the mesh, transformed by the prediction.

5.3. Low-Level Object Detector Evaluation

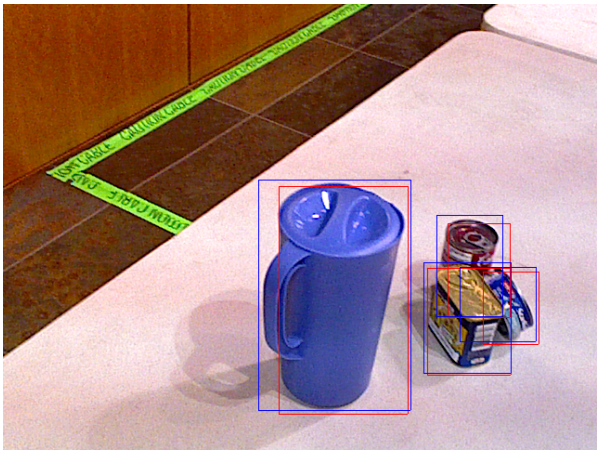


Figure 4. Visualization of one example output of the YOLOv5 object detector. The blue boxes are the ground-truth bounding boxes of the objects and the red boxes are the predicted outputs.

The YOLOv5 detector we use for the experiments was trained on a dataset that also uses YCB objects but it was not trained on the exact dataset as we use in our experiments. Therefore, there are classes that are never detected because they were not present in the dataset that was used for training the detector. For our further evaluations we just consider the 12 different object types that are recognized by the YOLOv5 detector.

Figure 4 shows the resulting predictions of the YOLOv5 detector on one image from the YCB-Video dataset and it can be seen that the correct regions are detected, but the predicted bounding boxes do not align perfectly with the ground truth boxes. Figure 5 is a visualization of the prediction of our low-level detector with 100 particles and 5 iterations per object.

Table 1 presents a detailed evaluation for the 12 object types

Object	Low Level Detector ADD	PHD Filter ADD	PoseCNN[1] ADD
002_master_chef_can	0.66	0.62	0.50
003_cracker_box	0.68	0.66	0.52
004_sugar_box	0.70	0.69	0.69
005_tomato_soup_can	0.65	0.65	0.66
006_mustard_bottle	0.52	0.52	0.79
007_tuna_fish_can	0.60	0.58	0.70
008_pudding_box	0.67	0.66	0.63
010_potted_meat_can	0.66	0.64	0.60
011_banana	0.69	0.67	0.72
025_mug	0.54	0.57	0.57
051_large_clamp	0.63	0.62	0.25
052_extra_large_clamp	0.54	0.54	0.15
Average	0.66	0.64	0.53

Table 1. Comparison of the ADD-score of our low-level detector, PHD-filtered detections and the PoseCNN (Xiang et al., 2018) on the YCB-Video dataset.

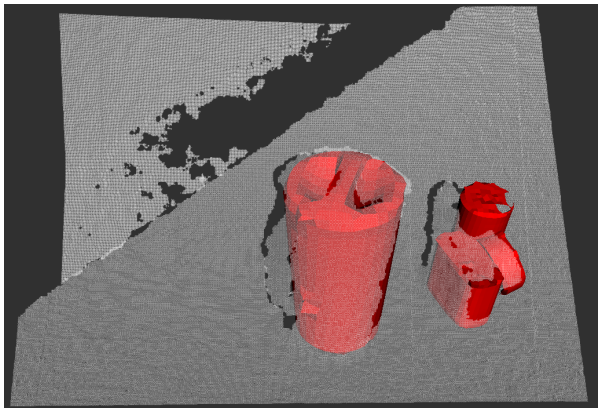


Figure 5. Visualization of the depth image, as a point cloud, and the resulting pose predictions of our low-level object detector.

in the YCB-Video dataset which are detected by our low-level detector. It can be seen that the overall performance of our probabilistic low-level detector is worse than the performance of the PoseCNN which is a pure deep-learning approach to pose estimation. The most noticable difference is in the performance on the "051_large_clamp" and "052_extra_large_clamp" classes. However, the performance of our approach on these two objects is very similar to the performance on all the other objects and the huge gap in performance between our approach and the PoseCNN comes from the fact that the PoseCNN performs very well on these types of objects.

It can also be seen that the performance of our approach is very similar for all the different object types present in the dataset. This shows that the object type does not matter for the performance of our low-level detector.

[1]The results for the PoseCNN experiments are the results reported in the PoseCNN publication (Xiang et al., 2018) trained on the YCB-Video dataset and tested on the same validation set as we use for our experiment.

5.4. High-Level Object Tracking Evaluation

Table 1 includes a detailed evaluation of our PHD filter. The input to the PHD filter are the detections of the low-level detector during evaluation. We see an overall improvement of 0.02 in terms of ADD compared to the unfiltered poses. This shows that the yield of the usage of the SMC-PHD filter is not too big, but it also shows that the PHD filter works for tracking the 6DoF poses of the detected objects.

6. Conclusion

In this work, we present an algorithm which is able to track multiple objects in the RGB-D images produced by depth cameras using probabilistic approaches and it demonstrates that the SMC-PHD filter can be used for 6DoF multi object tracking tasks.

A limiting factor in our work is the usage of YOLOv5 in the low-level detector because this approach does just detect objects it was trained on, while the other parts of our pipeline do just require a 3D model of the object without the necessity for any retraining. In a future work one might investigate if it is possible to use real-time segmentations of depth images like presented in (Abramov et al., 2012) and the recognition of the object type by using feature points of a 3D model and matching these with the image (Rothganger et al., 2006).

Another possible future work is to compare different variants of the PHD filter with each other to find the approach which works best for 6DoF pose tracking. The dynamic and measurement models used by the PHD filter could also be differentiated between the different target types.

References

Abramov, A., Pauwels, K., Papon, J., Wörgötter, F., and Dellen, B. Depth-supported real-time video segmentation with the kinect. In *IEEE Workshop on Applications of Computer Vision, WACV 2012, Breckenridge, CO, USA, January 9-11, 2012*, pp. 457–464. IEEE Computer Society, 2012. doi: 10.1109/WACV.2012.6163000. URL https://doi.org/10.1109/WACV.2012.6163000.

Çalli, B., Singh, A., Walsman, A., Srinivasa, S. S., Abbeel, P., and Dollar, A. M. The YCB object and model set: Towards common benchmarks for manipulation research. In *International Conference on Advanced Robotics, ICAR 2015, Istanbul, Turkey, July 27-31, 2015*, pp. 510–517. IEEE, 2015a. doi: 10.1109/ICAR.2015.7251504. URL https://doi.org/10.1109/ICAR.2015.7251504.

Çalli, B., Walsman, A., Singh, A., Srinivasa, S. S., Abbeel, P., and Dollar, A. M. Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols. *CoRR*, abs/1502.03143, 2015b. URL http://arxiv.org/abs/1502.03143.

Çalli, B., Singh, A., Bruce, J., Walsman, A., Konolige, K., Srinivasa, S. S., Abbeel, P., and Dollar, A. M. Yale-cmu-berkeley dataset for robotic manipulation research. *Int. J. Robotics Res.*, 36(3):261–268, 2017. doi: 10.1177/0278364917700714. URL https://doi.org/10.1177/0278364917700714.

Clark, D. E., Panta, K., and Vo, B. The GM-PHD filter multiple target tracker. In *9th International Conference on Information Fusion, FUSION 2006, Florence, Italy, July 10-13, 2006*, pp. 1–8. IEEE, 2006. doi: 10.1109/ICIF.2006.301809. URL https://doi.org/10.1109/ICIF.2006.301809.

Edman, V., Andersson, M., Granström, K., and Gustafsson, F. Pedestrian group tracking using the GM-PHD filter. In *21st European Signal Processing Conference, EUSIPCO 2013, Marrakech, Morocco, September 9-13, 2013*, pp. 1–5. IEEE, 2013. URL http://ieeexplore.ieee.org/document/6811759/.

García, F., Prioletti, A., Cerri, P., and Broggi, A. PHD filter for vehicle tracking based on a monocular camera. *Expert Syst. Appl.*, 91:472–479, 2018. doi: 10.1016/j.eswa.2017.09.018. URL https://doi.org/10.1016/j.eswa.2017.09.018.

Issac, J., Wüthrich, M., Cifuentes, C. G., Bohg, J., Trimpe, S., and Schaal, S. Depth-based object tracking using a robust gaussian filter. In Kragic, D., Bicchi, A., and Luca, A. D. (eds.), *2016 IEEE International Conference on Robotics and Automation, ICRA 2016, Stockholm, Sweden, May 16-21, 2016*, pp. 608–615. IEEE, 2016. doi: 10.1109/ICRA.2016.7487184. URL https://doi.org/10.1109/ICRA.2016.7487184.

Jocher, G., Stoken, A., Borovec, J., NanoCode012, ChristopherSTAN, Changyu, L., Laughing, tkianai, yxNONG, Hogan, A., lorenzomammana, AlexWang1900, Chaurasia, A., Diaconu, L., Marc, wanghaoyang0106, ml5ah, Doug, Durgesh, Ingham, F., Frederik, Guilhen, Colmagro, A., Ye, H., Jacobsolawetz, Poznanski, J., Fang, J., Kim, J., Doan, K., and Yu, L. ultralytics/yolov5: v4.0 - nn.SiLU() activations, Weights & Biases logging, PyTorch Hub integration. January 2021. doi: 10.5281/zenodo.4418161. URL https://doi.org/10.5281/zenodo.4418161.

Mahler, R. P. S. Multitarget bayes filtering via first-order multitarget moments. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1152–1178, 2003. doi: 10.1109/TAES.2003.1261119.

Markley, F. L., Cheng, Y., Crassidis, J. L., and Oshman, Y. Averaging quaternions. *Journal of Guidance, Control, and Dynamics*, 30(4):1193–1197, 2007.

Rothganger, F., Lazebnik, S., Schmid, C., and Ponce, J. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. In *International Journal of Computer Vision*, pp. 231–259. 2006. doi: 10.1007/s11263-005-3674-1. URL https://doi.org/10.1007/s11263-005-3674-1.

Schikora, M., Gning, A., Mihaylova, L., Cremers, D., and Koch, W. Box-particle PHD filter for multi-target tracking. In *15th International Conference on Information Fusion, FUSION 2012, Singapore, July 9-12, 2012*, pp. 106–113. IEEE, 2012. URL http://ieeexplore.ieee.org/document/6289793/.

Shoemake, K. Iii.6 - uniform random rotations. In KIRK, D. (ed.), *Graphics Gems III (IBM Version)*, pp. 124–132. Morgan Kaufmann, San Francisco, 1992. ISBN 978-0-12-409673-8. doi: https://doi.org/10.1016/B978-0-08-050755-2.50036-1. URL https://www.sciencedirect.com/science/article/pii/B9780080507552500361.

Song, Y. and Jeon, M. Online multiple object tracking with the hierarchically adopted gm-phd filter using motion and appearance. In *2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, pp. 1–4, 2016. doi: 10.1109/ICCE-Asia.2016.7804800.

Vo, B. . and Ma, W. . The gaussian mixture probability hypothesis density filter. *IEEE Transactions on Signal Processing*, 54(11):4091–4104, 2006. doi: 10.1109/TSP.2006.881190.

Vo, B. ., Singh, S., and Doucet, A. Sequential monte carlo methods for multitarget filtering with random finite sets. *IEEE Transactions on Aerospace and Electronic Systems*, 41(4):1224–1245, 2005. doi: 10.1109/TAES.2005.1561884.

Wuthrich, M., Pastor, P., Kalakrishnan, M., Bohg, J., and Schaal, S. Probabilistic object tracking using a range camera. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, November 3-7, 2013*, pp. 3195–3202. IEEE, 2013. doi: 10.1109/IROS.2013.6696810. URL https://doi.org/10.1109/IROS.2013.6696810.

Wuthrich, M., Cifuentes, C. G., Trimpe, S., Meier, F., Bohg, J., Issac, J., and Schaal, S. Robust gaussian filtering using a pseudo measurement. In *2016 American Control Conference, ACC 2016, Boston, MA, USA, July 6-8, 2016*, pp. 3606–3613. IEEE, 2016. doi: 10.1109/ACC.2016.7525473. URL https://doi.org/10.1109/ACC.2016.7525473.

Xiang, Y., Schmidt, T., Narayanan, V., and Fox, D. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In Kress-Gazit, H., Srinivasa, S. S., Howard, T., and Atanasov, N. (eds.), *Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018*, 2018. doi: 10.15607/RSS.2018.XIV.019. URL http://www.roboticsproceedings.org/rss14/p19.html.