iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** I **Month of publication:** January 2021

DOI: https://doi.org/10.22214/ijraset.2021.32890

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue I Jan 2021- Available at www.ijraset.com

Dynamic Object Elimination based Indoor 3D Mapping System

Pritam Mane[1], Vedant Kumar[2], Sreekar L[3], Pallavi Malame[4], Reena Sonkusare[5], Sukanya Kulkarni[6]
[1, 2, 3, 4, 5, 6]*Department of Electronics and Telecommunication, Bharatiya Vidya Bhavans Sardar Patel Institute of Technology*

Abstract: ***In today's world, the need for autonomous robots is increasing at an exponential rate and the implementation of Simultaneous Localisation And Mapping (SLAM) is gaining more and more attention. One of the major component of SLAM is 3D Mapping of the environment which enables autonomous robots to perceive the environment like a human does for which many Depth cameras or RGB-D cameras prove useful. This paper proposes a continuous real-time 3D mapping system that tackles the long existing problem of point cloud distortion induced by dynamic objects in the frame. Our method uses the Microsoft Kinect V1 as the RGB-D camera and the packages in the Robotic Operating System (ROS) like the Real Time Appearance Based Mapping (RTAB-map) for 3D reconstruction. A ROS based method is used to implement dynamic object elimination in real-time. For the purpose of dynamic objects detection in the frame, two algorithms - Deep Learning based tiny YOLO-v3 and a Machine Learning based Haar Cascade classifier are used. The results from the two are compared in terms of accuracy, execution time and mean Average Precision (mAP) and it was inferred that although Haar Cascade model is comparatively less accurate when detecting objects, it is two times faster than YOLO which makes the system more real-time. The real-time implementation was given more preference while selecting the model.***

Keywords: Simultaneous Localization and Mapping(SLAM), 3D mapping, Robot Operating System (ROS), Deep learning, Stereo Vision, Object Detection, Point Cloud, Scale invariant feature transform (SIFT)

I. INTRODUCTION

In today's world where the need of autonomous robots is ever increasing, the implementation of Simultaneous Localisation And Mapping (SLAM) is gaining more and more attention. Robotic cleaners and inventory management robots are autonomous and have the same functionality in both the cases, that is, to navigate and find their way through an unknown environment. In order to do so, they need to have a frame of reference and perceive the nearby environment (through computer vision or through Lidar sensors) to sketch a map of its surroundings and traverse its path. 3D reconstruction involves using computer vision to perceive the external surroundings an building a depth based 3D point cloud. Hence, the underlying principle is a common intersection point between 3D reconstruction and autonomous navigation. Autonomous robot navigation is a step ahead of 3D reconstruction. The rise in demand for 3D solutions calls for a holistic solution that can perceive the surroundings around and create a 3D map of the same.

The usage of computer vision for robotic automation and programming robots or creating 3D layouts has been quite prevalent. The SLAM (simultaneous localization and mapping) problem has persisted for a long time and a lot of research done portrays new ways and methodologies to tackle the problem for mapping. Robot navigation has long been an important and active area of research.

Existing innovations in this field utilize expensive cameras for creating depth and disparity maps which tend to be accurate, but the overall cost of the product increases. Other techniques include using stereo vision cameras to calculate the depth which can further be used to create 3D point clouds. This method of reconstruction is fairly cheaper but accuracy is compensated. The existing systems work very well for reconstruction of static frames but when dynamic objects enter the frame the output is not very accurate. Our system proposes a method for detecting and removing the dynamic objects from the frame in order to create an accurate 3D reconstruction of a scene.

The pipeline for the 3D reconstruction consists of 4 major steps:

1) Preparing the camera system used for retrieving image and depth data
2) Creation of depth and disparity maps
3) Generating and aligning point clouds
4) Refining and processing the point clouds

Active research has been done in all the domains. The methods for 3D reconstruction and point cloud generation using stereo vision or depth estimation using Lidars have been implemented. Novel methods for depth estimation using Convolutional Neural networks have been previously performed. A work we referred, describes a unique depth sensing approach using convolutional neural networks (CNN)[1]. This falls under monocular SLAM vision which uses only a single camera, maybe coupled with other sensors for depth estimation. Although this eliminates the need of RGB-D cameras the process gives high errors.

Another traditional method is Depth Estimation using Stereo Vision which works on the principal of triangulation [2,3]. Since the images are obtained from 2 different cameras in a stereo system, feature matching algorithms are needed in order to synchronize the two cameras in the stereo system [4]. Instead of using a stereo vision based camera system an RGB-D camera can also be used. This takes away the effort in aligning and synchronizing the two cameras in the system [5].

The framework for implementing the 3D reconstruction or the SLAM algorithm, is much more viable and is less prone to errors in accuracy [6]. One such RGB-D camera system is the Microsoft Kinect. The library - Kinect Fusion or KinFu is compatible with many RGB-D camera systems and helps in processing of the point cloud. [7,8,9]. KinFu is an open source library containing many functions to use for 3D reconstruction[10,11].

In order to align multiple point clouds for a complete 3D reconstruction, feature matching algorithms like SIFT are used [12]. The process of stitching multiple point clouds same time during the 3D reconstruction of a particular frame can be computationally expensive and thus may not give real time performance. Dimensionality reduction methods like SVD for point cloud can be used. Another work we referred, proposes a method of 2d feature matching and hash index lookup for embedding the new point cloud onto another for faster computation[13]. ROS (Robotic operating system) provides with a few packages to visualise the developed map. This makes it easy to integrate and use the RGB-D camera for robotic applications. The usage of the ROS platform and leveraging its packages for simulation and visualization streamlines the process of 3D reconstruction [14]. A package very widely used for generating 3D point clouds of the environment for navigation is the RTAB-Map [15]. In order to make things computationally less expensive the fast Sampling Plane Filtering (FSPF) can be used to reduce the volume of the 3D point cloud by sampling points from the depth image [16]. Denoising methods can be used to improve the accuracy of the 3D reconstruction but that improves the accuracy only to a certain extent. Another method is to filter out the samples of dynamic objects from the input data. The image data from the kinect sensor is scanned and the dynamic objects can be filtered out. This can be done using deep learning models in order to detect the dynamic objects [17,18,19].

All the research work based on mapping and 3D reconstruction are very extensive and constructive and provide accurate results when static objects present in the frame are reconstructed. However, when dynamic objects are present in the frame, the qualitative results of the 3D reconstruction are affected. The presence of dynamic objects in the frame during the process of reconstruction makes the reconstruction noisy. In this research work we have proposed a solution to detect and eliminate dynamic objects (like people) from the frame and thus prevent them from being reconstructed in the 3D point cloud. This makes the 3D map generated immune to the distortion caused by dynamic objects in the environment.

II. METHODOLOGY

Our proposed solution is an RGB-D sensor based medium to long range 3D reconstruction system that dynamically detects and removes objects present in the frame in order to produce an accurate 3D reconstruction. The RGB-D camera we have used is a Microsoft Kinect v1 camera, it estimates the depth using an IR sensor and the output obtained consists of the image data (the RGB values) and the depth data (distance from the of the object from the camera). The depth data obtained from the camera was validated by generating the depth map. This shows the relative distance between the objects in the frame and the camera. The image data, that's obtained from the camera flows to the object detection node which detects the dynamic objects present in the frame. The frames in which dynamic objects are detected are discarded and the remaining frames are moved forward for the final 3D reconstruction. Our system prevents the loss of loss of data and the noisiness that is induced when an object enters the frame at a point and leaves at another point.

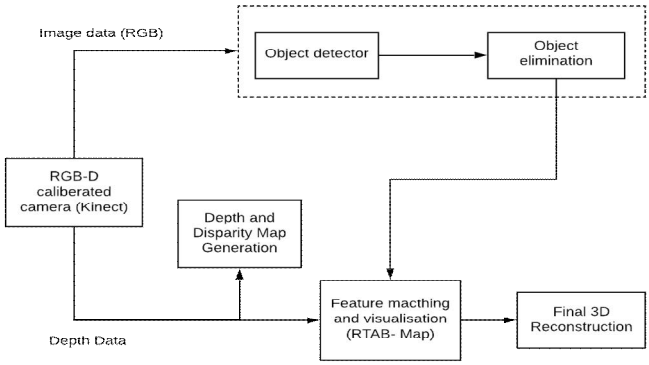


Fig. 1 System Block Diagram

III.RGB-D SENSOR & 3D RECONSTRUCTION

The Microsoft Kinect camera has 4 intrinsic parameters: two parameters related to focal length are denoted as fx_d and fy_d. The remaining two color-related parameters are represented as cx_d and cy_d. These parameters are vital in converting the raw coordinates into the real world coordinates. The Kinect V1 can measure the raw depth in the range of 0 to 2047. To calculate raw depth into meters, the following formula [20] is employed.

$$\text{depth} = 1 / ((\text{raw depth value}) \times (-0.003 + 3.330))$$

Similarly, to convert the x and y coordinates of a pixel in a frame into real world coordinates, following values of intrinsic parameters are utilized [20]

$$fx\_d = 1 / (5.94e + 02)$$
$$fy\_d = 1 / (5.91e + 02)$$
$$cx\_d = 3.39e + 02$$
$$cy\_d = 2.42e + 02$$

The real world x and y coordinates are as follows-

$$x = ((x - cx\_d) \times \text{depth} \times fx\_d)$$
$$y = ((y - cy\_d) \times \text{depth} \times fy\ d)$$
$$z = \text{depth}$$

The RTAB-Map uses a global loop closure detection technique for generating 3D point clouds. SIFT algorithm was used for feature matching. The SIFT algorithm works in four steps - Scale Space construction, localisation of key points, Orientation Assignment, and Key point Descriptor. The following steps and formulas are involved in the corresponding steps [21]. In the first step, Gaussian blur function is used to blur images of various scales.

$$L(x, y, \sigma) = G(x, y, \sigma) \times I(x, y)$$

Where G(x,y,σ) is the Gaussian filter convolved over the image I(x,y) to obtain the output as L(x,y,σ) Following this, the original image is subtracted with their next blurred image as a feature enhancement step.
Final output of this step would be:

$$D(x, y, \sigma) = I(x, y) - G(x, y, \sigma) * I(x, y)$$

The formula/difference of Gaussian can be generalised for any consecutive blurred images as:

$$G(x, y, k) - G(x, y, \sigma) * I(x, y)$$

This step is followed by computing local maxima or local minima of the images. The pixels so selected are the potential key points that are scale invariant. By using SIFT for correspondence matching, we come to know the number of matching key points between the two consecutive or non-consecutive frames. RTAB-Map has features to visualise and match these key points. If the number of key points matched are above a certain set threshold, the entity is regarded as the same

IV.ROS IMPLEMENTATION

The robot operating system or ROS comprises of various nodes through which communication takes place. Each sensor or component in the system which generates or utilizes data will be assigned a node. The communication between these nodes (and thus among the sensors) happens over a topic.
For information to be passed between the two nodes, they should be connected to the same topic. A node can act as a publisher which publishes or sends the data onto the topic or as a subscriber which receives the data from the topic. The user, however, can leverage the autonomy of creating custom nodes to publish or subscribe to a topic. This is what allows the processes in autonomous robots to co-ordinate with each other.
In our system, the two nodes used as standard nodes are the Kinect camera and the RTAB-Map. Kinect publishes the data on some topics which is received by the RTAB-Map by subscribing to those topics. The Kinect camera publishes three information - RGB image, depth information, and camera information. The RTAB-Map receives this data, generates the map data and further publishes it to rtabmapviz for final visualisation. This is illustrated in Figure [2].

For point cloud generation, the color image frame obtained from the Kinect has an RGB-A format and depth of each pixel is needed to map it on the point cloud. To compute the depth value for a given pixel (x,y), an index is calculated with the formula:

$$\text{index} = x + (\text{width} \times y)$$

This index value in the depth array is the depth value of the pixel (x,y). Finally, the byte-valued (0-255) RGB-A format is converted into a float-valued (0.0-1.0) RGB format.

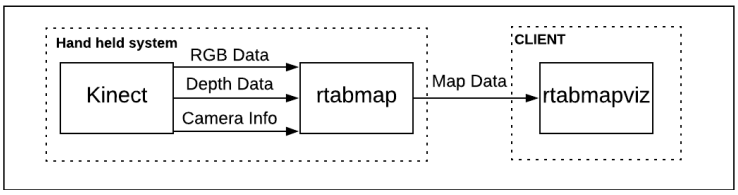


Fig. 2. ROS node architecture for 3D Reconstruction

The proposed solution aims to tackle the problem of the projection of dynamic objects on the RTAB-Map. These dynamic moving objects tend to hinder the accurate reconstruction of the surrounding. For the same reason, it is imperative to build a solution that counters this problem. Therefore, the solution introduces a custom node between the Kinect camera and the RTAB-Map. Our node named Object Elimination acts as a middle substitution between the Kinect Node and the RTAB-Map Node. So now the data published by the Kinect is first fetched by out node. Then the object detection algorithm is used to detect a person in the RGB frame only. Figure [3] shows the new architecture with our node included. If a person is detected, that particular RGB frame and the corresponding depth frame is skipped or discarded. In this way, the reconstructed scene will purely be of static objects. Figure [4] depicts this algorithm.

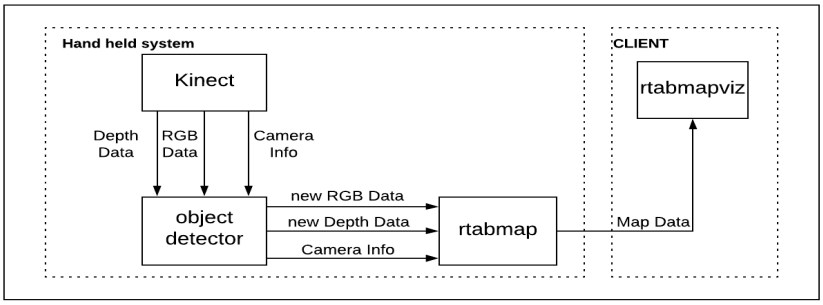


Fig. 3. Custom node architecture

In terms of ROS, the nodes that previously published the image data to RTAB-Map will now be published to the object detector node. But, the format that ROS uses to transfer images to the custom node is different from the standard PNG or JPEG format. The sensor_msgs/Image message format, therefore, has to be converted in a format that can be processed by libraries like OpenCV. For this, the cvbridge() function is used to convert the format of the image into the format compatible with ROS. The object detection algorithm is then applied on this JPEG image.

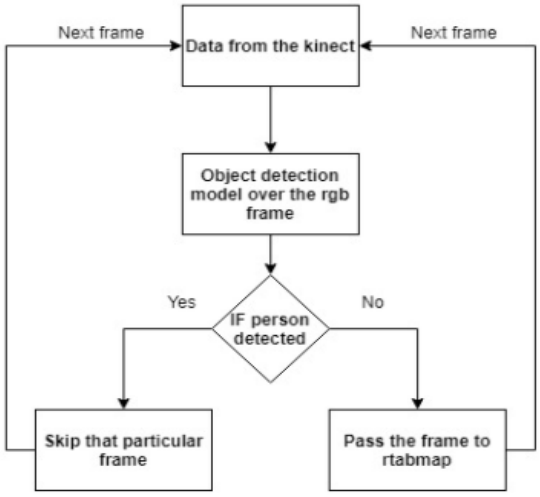


Fig. 4. Flowchart for 3D Mapping with Dynamic Environment

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue I Jan 2021- Available at www.ijraset.com

V. OBJECT ELIMINATION ALGORITHM

Classifying objects as dynamic is a subjective dilemma and depends on the problem involved. For indoor applications, humans can be considered as dynamic objects. For a self-driving car, humans, cars, and other vehicles can be considered as dynamic objects, Therefore, the model employed should be trained on the basis of the application involved.

A. Deep Learning Based Approach

A Convolutional Neural Network (CNN) based Deep Learning model, tiny YOLOv3[22] is used for dynamic object. The technique involves object classification and localization. If the probability of the presence of dynamic class is above a certain threshold, a bounding box is drawn around the object. The probability score is usually assigned by the Softmax function or kernel SVM (Support vector machine). In our case, dynamic object is the person.

But, if the model has to detect the dynamic objects it hasn't been trained on, either a custom object detector can be made, or a technique called transfer learning can be used. However, the model that we used which is tiny YOLOv3 for people detection proved to be slow. By the time YOLO was computing object detection algorithm on a single frame, other incoming frames were published to the RTAB-Map.

B. Haar Cascade Based Approach

As the deep learning based solution was too slow in computation to be used real-time, a simpler Haar Cascade based model [23] was used. Although the model is not as accurate as the CNN based object detector, it is cheaper in terms of computation power and hence can be deployed real-time. Haar Cascade is a simple Haar feature based cascade classifier. The process involves convolving the input image with multiple filters. Initial filters detects contours and edges in all directions. Other filters are applied to extract features for detecting a specific class. Another advantage of this algorithm is that it is trained specifically for detecting body parts and face and hence only specific weight files can be used to detect corresponding class. This ensures lesser computation as compared to YOLO which is trained to detect 80 different classes. Also to fasten the overall process, the proposed solution skips the frames based on the detection of human faces.

Once the image is processed, it has to be further transmitted to the forward node. For this, the OpenCV image has to be again converted in a ROS image message format. The CV format and ROS image message must always have the same number of channels and pixel depths. The custom node passes the processed image information in a custom generated topic. To this topic, the RTAB-Map must be subscribed for the reconstruction of the surrounding.

VI. RESULTS AND ANALYSIS

The 3D reconstruction is done using Kinect drivers and RTAB-Map on ROS. To verify that our system is working we considered 3 main point clouds of the same scene (environment). Figure [5] shows the normal point cloud of the room when our custom Object Elimination node is inactive and also there are no dynamic objects in the room.

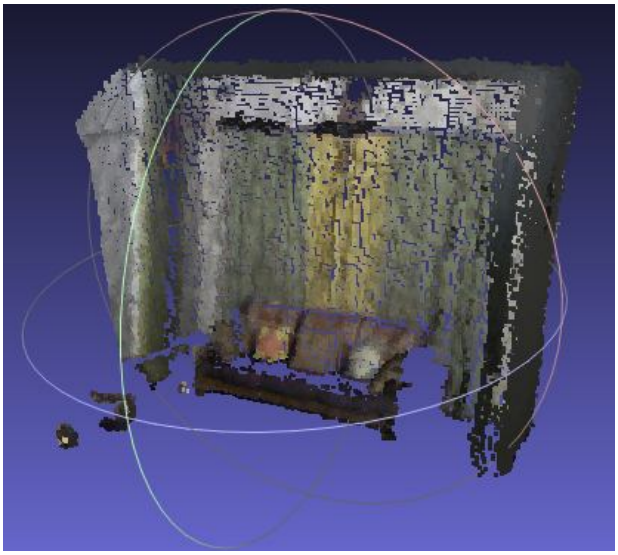


Fig. 5. 3D Reconstruction of a room using a Microsoft Kinect camera

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue I Jan 2021- Available at www.ijraset.com

The other two point clouds are recorded when there is a dynamic object (person) present in the frame. Figure [6] shows the point cloud where our Object detection node is inactive and a dynamic object is present in the frame. It can be seen that the point cloud is being distorted as the person is in front of the camera as the depth and RGB data of the person is also recorded in the point cloud.

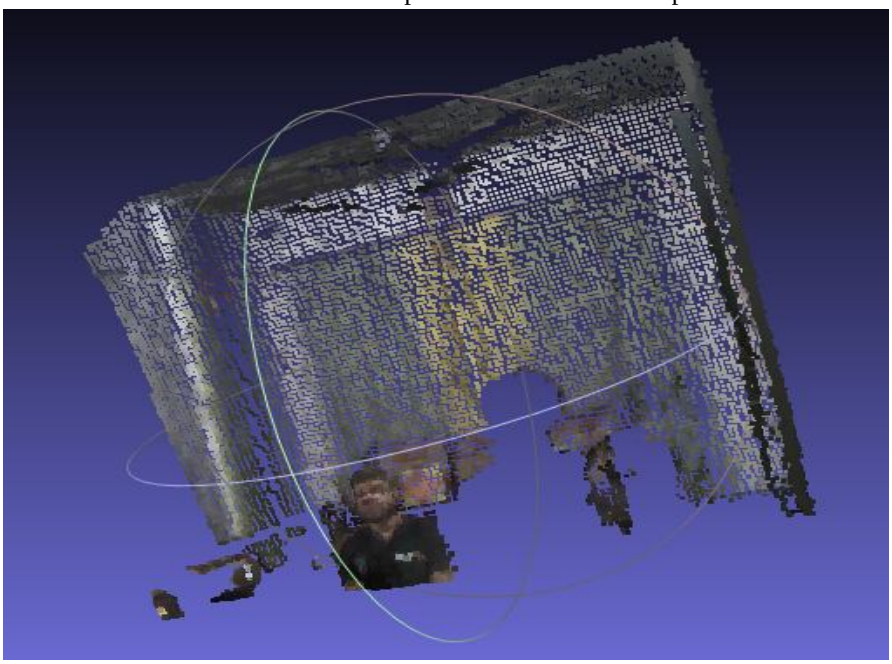


Fig.6. 3D Reconstruction with Object Elimination node inactive

Figure [7] shows the point cloud registered when our Object Elimination Node was active and there was a dynamic object in the frame. As it can be seen, the point cloud is not distorted because our node has discarded the frame in which a person's face was detected.

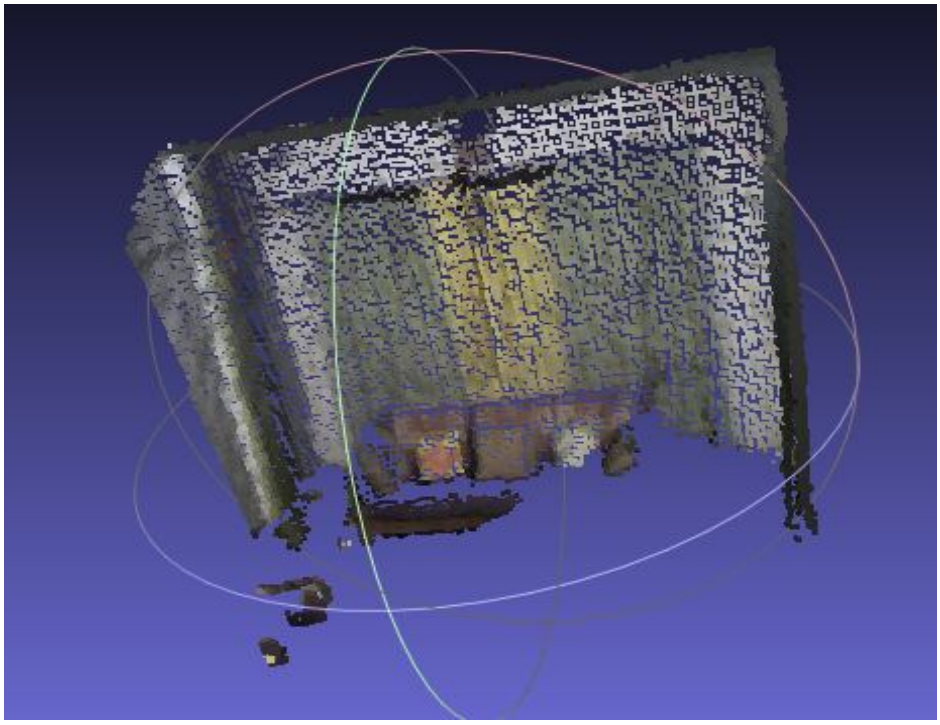


Fig. 7. 3D Reconstruction with Object Elimination node active

Further the point clouds of Figure [5] and Figure [6] were compared in Matlab to find the similar features and the difference between them. Figure [8] shows the output of this comparison. The green part is the extra part in the point cloud where our Node was inactive. It is evident that the green part is caused due to the dynamic object (person).

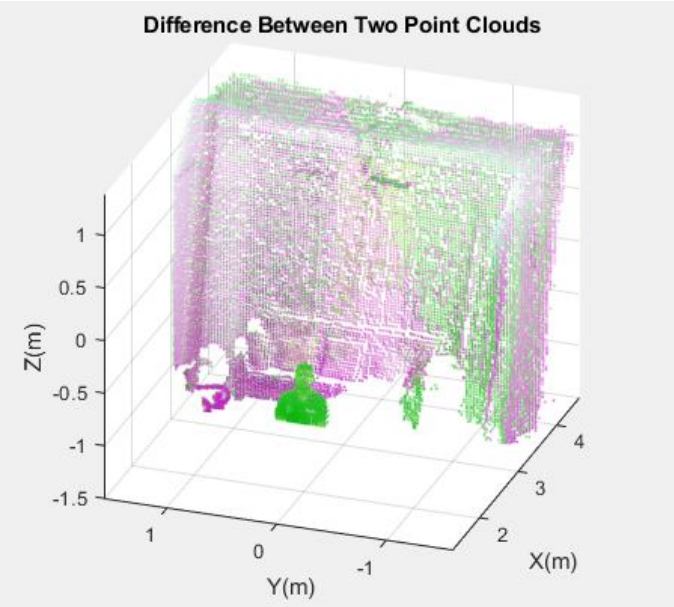


Fig. 8. Difference between the Point Clouds

Table I
Comparison of model parameters for two different models

Object Detection Model	Accuracy	Run time	mAP
Tiny YOLO v3 (CNN Based)	87%	200 msec	63%
Haar Cascade	65%	84 msec	49%

Table 1 demonstrates the performance of two classification models used - YOLOv3 and Haar Cascade. There is tradeoff between the accuracy and the run time of the models. For efficient reconstruction to take place, it was crucial for the model to have minimum time complexity. For the same reason, the proposed solution incorporates Haar Cascade based classifier for detecting dynamic objects like humans. Table 2 shows the comparison of the number of points in both the point clouds. It can be inferred that when the object elimination algorithm is implemented the number of points reduce, that is, the dynamic objects are not being reconstructed. This verifies the working of our methodology to remove dynamic objects from the reconstruction of a particular frame.

Table II
Number Of Points In Point Cloud Analysis

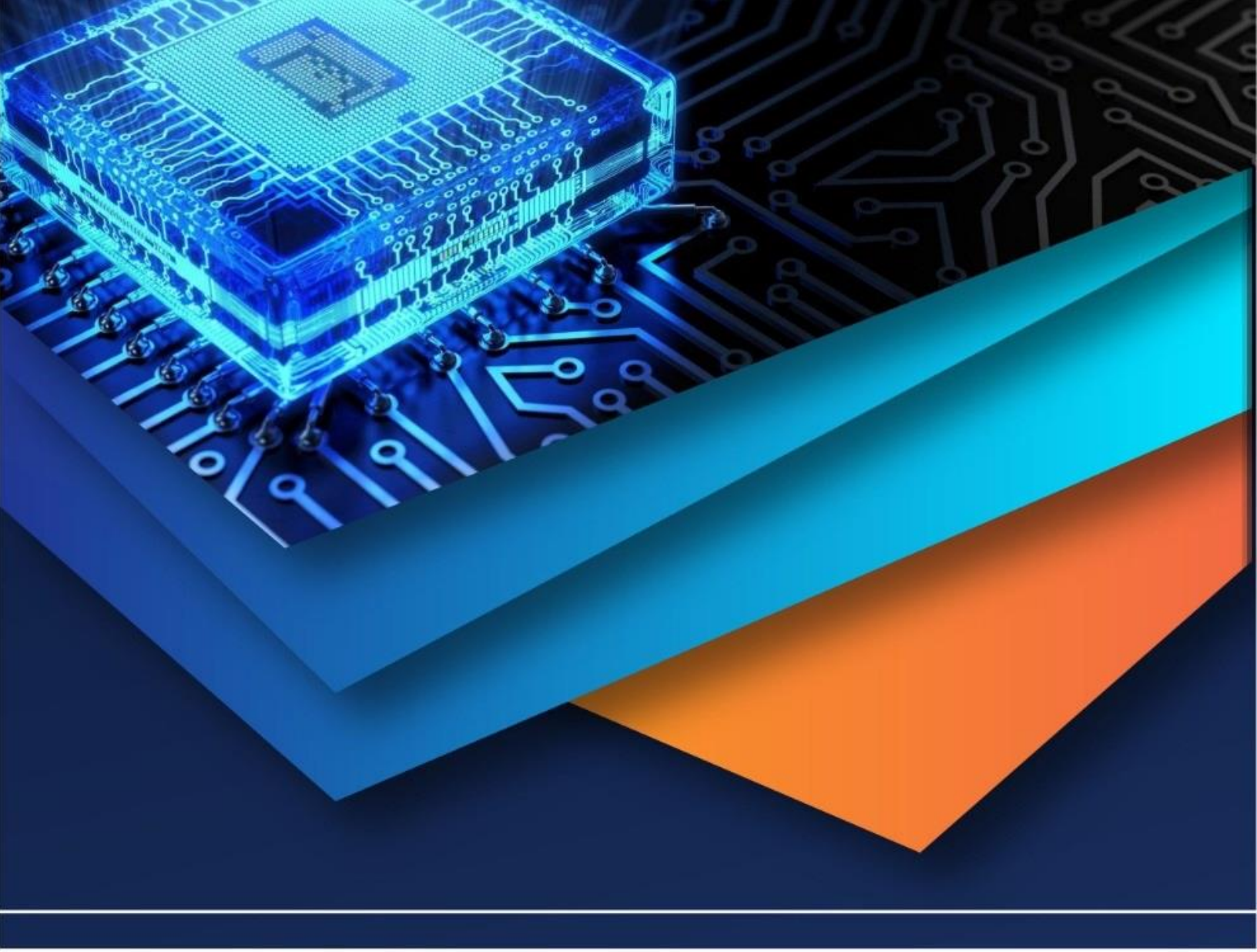
With Object elimination algo	Without object elimination algo
14219	14865

VII. CONCLUSION

This paper demonstrates a methodology to improve the qualitative accuracy of the point cloud formed in a dynamic environment by detecting and eliminating the dynamic objects, specifically human beings. It compares two object detection algorithms, tiny YOLOv3 model and Haar Cascade model, in terms of accuracy and computational complexity. Although YOLO gives accurate results, it's accuracy is affected to a great extent while deploying real-time on our 4GB GPU based system. The Haar Cascade model is computationally less expensive and provides more faster real-time results. The average execution time of YOLO is 200ms per frame as compared to the 84ms of Haar Cascade. The scope of this paper can be extended by performing image segmentation. Instead of discarding the frames that contain the dynamic objects, the images can be segmented before constructing the point cloud. This will improve the accuracy of the final 3D reconstruction.

REFERENCES

[1] K. Tateno, F. Tombari, I. Laina and N. Navab, "CNN-SLAM: RealTime Dense Monocular SLAM with Learned Depth Prediction," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6565-6574, doi: 10.1109/CVPR.2017.695.
[2] B. Pal, S. Khaiyum and Y. S. Kumaraswamy, "3D point cloud generation from 2D depth camera images using successive triangulation," 2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bangalore, 2017, pp. 129-133, doi: 10.1109/ICIMIA.2017.7975586.
[3] P. R. Induchoodan, M. J. Josemartin and P. R. Geetharanjin, "Depth recovery from stereo images," 2014 International Conference on Contemporary Computing and Informatics (IC3I), Mysore, 2014, pp. 745-750, doi: 10.1109/IC3I.2014.7019764.
[4] M. Svedman, L. Goncalves, N. Karlsson, M. Munich and P. Pirjanian, "Structure from stereo vision using unsynchronized cameras for simultaneous localization and mapping," 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, Edmonton, Alta., 2005, pp. 3069-3074, doi: 10.1109/IROS.2005.1545431.
[5] Henry, Peter Krainin, Michael Herbst, Evan Ren, Xiaofeng and Fox, Dieter. (2012). RGB-D Mapping: Using Kinect-Style Depth Cameras for Dense 3D Modeling of Indoor Environments. International Journal of Robotic Research - IJRR. 31. 647-663. 10.1177/0278364911434148
[6] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.
[7] I. C. Patino Mejia and A. Zell, "3D Reconstructions with KinFu Using ˜ Different RGBD Sensors," 2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS), Sophia Antipolis, France, 2018, pp. 13-18, doi: 10.1109/IPAS.2018.8708892.
[8] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real- time dense surface mapping and tracking," in IEEE Intl. Symposium on Mixed and Augmented Reality, Oct 2011, pp. 127–136.
[9] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," IEEE International Conference on Robotics and Automation, pp. 1524–1531, 2014.
[10] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGBD SLAM systems," IEEE International Conference on Intelligent Robots and Systems, pp. 573–580, 2012.
[11] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGBD SLAM systems," IEEE International Conference on Intelligent Robots and Systems, pp. 573–580, 2012.
[12] Chu, Jun Nie, Chun-mei. (2011). Multi-view point clouds registration and stitching based on SIFT feature. 1. 10.1109/ICCRD.2011.5764019.
[13] L. He, Z. Li and S. Chen, "Aligning Algorithm of 3D Point Cloud Model Based on Dimensionality Reduction," 2017 2nd International Conference on Multimedia and Image Processing (ICMIP), Wuhan, 2017, pp. 281- 285, doi: 10.1109/ICMIP.2017.34.
[14] Takeda, H. (2012). Study on the Indoor SLAM Using Kinect. In Advanced Methods, Techniques, and Applications in Modeling and Simulation (pp. 217–225). Springer Japan.
[15] Labbe, M. and Michaud, F., 2018. "Rtab-map as an open-source lidar ´ and visual slam library for large-scale and long-term online operation". Journal of Field Robotics.
[16] J. Biswas and M. Veloso, "Depth camera based indoor mobile robot localization and navigation," 2012 IEEE International Conference on Robotics and Automation, Saint Paul, MN, 2012, pp. 1697-1702, doi: 10.1109/ICRA.2012.6224766.
[17] Soares, Joao and Gattass, Marcelo and Meggiolaro, Marco. ˜ (2019). MAPPING IN DYNAMIC ENVIRONMENTS USING DEEP LEARNING-BASED HUMAN DETECTION. 10.26678/ABCM.COBEM2019.COB2019-0845.
[18] H. He, S. Wang and S. Ma, "Research on Object Location Detection based on ROS Machine Vision," 2019 Chinese Control And Decision Conference (CCDC), Nanchang, China, 2019, pp. 1008-1012, doi: 10.1109/CCDC.2019.8833232.
[19] Tan, W., Liu, H., Dong, Z., Zhang, G. and Bao, H., 2013. "Robust monocular slam in dynamic environments". In IEEE International Symposium on Mixed and Augmented Reality (ISMAR)
[20] Z. Zhang, "A Flexible New Technique for Camera Calibration," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 11, 2000, pp. 13301334.
[21] Zhu Daixian, "SIFT algorithm analysis and optimization," 2010 International Conference on Image Analysis and Signal Processing, Zhejiang, 2010, pp. 415-419, doi: 10.1109/IASP.2010.5476084.
[22] Redmon, J., Ali, F.: YOLO9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7263–7271 (2017)
[23] Adolf, F. How-to build a cascade of boosted classifiers based on Haarlike features.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY