CONFIDENTIAL. Limited circulation. For review only.

A Light Field Front-end for Robust SLAM in Dynamic Environments

Pushyami Kaveti[1] and Hanumant Singh[2]

***Abstract*— There is a general expectation that robots should operate in urban environments often consisting of potentially dynamic entities including people, furniture and automobiles. Dynamic objects pose challenges to visual SLAM algorithms by introducing errors into the front-end. This paper presents a Light Field SLAM front-end which is robust to dynamic environments. A Light Field captures a bundle of light rays emerging from a single point in space, allowing us to see through dynamic objects occluding the static background via Synthetic Aperture Imaging(SAI). We detect apriori dynamic objects using semantic segmentation and perform semantic guided SAI on the Light Field acquired from a linear camera array. We simultaneously estimate both the depth map and the refocused image of the static background in a single step eliminating the need for static scene initialization. The GPU implementation of the algorithm facilitates running at close to real time speeds of 4 fps. We demonstrate that our method results in improved robustness and accuracy of pose estimation in dynamic environments by comparing it with state of the art SLAM algorithms.**

I. Introduction

SLAM enables robot navigation in unknown environments by accurately estimating the robot's pose and building the map of the world at the same time. It is essential for collision-free navigation in various indoor and outdoor robotic applications. Most of the research in SLAM assumes a static world, even though the robots are expected to function in predominantly dynamic environments. The static world assumption holds true for some applications during small scale and short-term runs, but limits the robots from operating in the real-world. The presence of dynamic entities like people, automobiles and bicycles in the real world causes errors in various stages of SLAM if these objects are not detected and dealt with. Feature matches associated with dynamic objects result in initialization failures, errors in pose estimation and incorrect maps of the world. If the dynamic features are included in the map they affect loop closure and re-localization preventing re-usability of maps for long-term applications. All these problems have motivated researchers to develop algorithms that target dynamic environments [1][2].

Dynamic environments are usually handled by detecting dynamic content in a variety of ways [3][4][5][6][7][8] and explicitly omitting the features associated with the detected dynamic objects while performing SLAM. Discarding information this way may result in errors if the dynamic content occupies a significant portion of the image or dominates the scene in terms of feature space. Therefore, extracting static features is imperative to estimating the pose of the robot accurately.

In this paper, we introduce a Light Field front-end built on top of ORBSLAM2[3] that is robust to dynamic environments. The front-end aims at accurate pose estimation and builds the static map of the world for long term use. While Light Fields have been researched extensively in the computer graphics community, their applicability to mobile robotics is far more recent[9][10][11]. A Light Field captures spatial and angular radiance in the form of a bundle of rays coming from a point in space as opposed to a single ray in a monocular camera. This helps us to extract the rays emerging from partially occluded portions of the scene via Synthetic Aperture Imaging (SAI).

We apply SAI to the Light Field collected from a camera array to synthesize the refocused image of the static background occluded by dynamic objects which is then passed on to extract features for tracking. We rely on deep learning based semantic segmentation for detecting a priori dynamic objects. We use the dynamic object detections to guide the SAI and recover the static background image essentially seeing through the dynamic objects as if they were never present at all. Thus, our method is effective even when the dynamic objects dominate the scene. Detecting a priori dynamic objects supports long term mapping when the objects are temporarily static and eliminates the need for initialization if they are actively moving. Our GPU implementation allows us to run the front-end processing close to real-time. In the next sections we discuss the Light Field front-end in detail and evaluate it on real datasets.

II. Related Work

SLAM in Dynamic Environments A large body of recent research addresses dynamic scenes by using optical flow [6], RANSAC[3], or residual cost functions [5][12] to detect and filter dynamic outliers. Outlier rejection strategies fail when the dynamic content constitutes a significant fraction of the scene. Other algorithms enforce temporal and structural consistency to detect dynamic content and omit such content from tracking and mapping. In [13], the background is reconstructed by accumulating warped depth between consecutive RGB-D images. [4] track changes by projecting sparse map points into the current frame. StaticFusion [12] detects moving objects and fuses temporally consistent data via model alignment. Temporal methods require initialization of static map in dynamic environments. We use CNN based semantic segmentation[14][15], which has emerged as an efficient instantaneous approach to detect apriori dynamic objects in many recent works[8][7][16][17][18]. Unlike

[1]Pushyami Kaveti is a student at Khoury College of Computer Sciences, Northeastern University, Boston kaveti.p@husky.neu.edu

[2]Hanumant Singh is with the Department of Electrical Engineering,Northeastern University ha.singh@northeastern.edu

Manuscript 1563 submitted to 2021 IEEE International Conference on Robotics and Automation (ICRA). Received October 31, 2020.

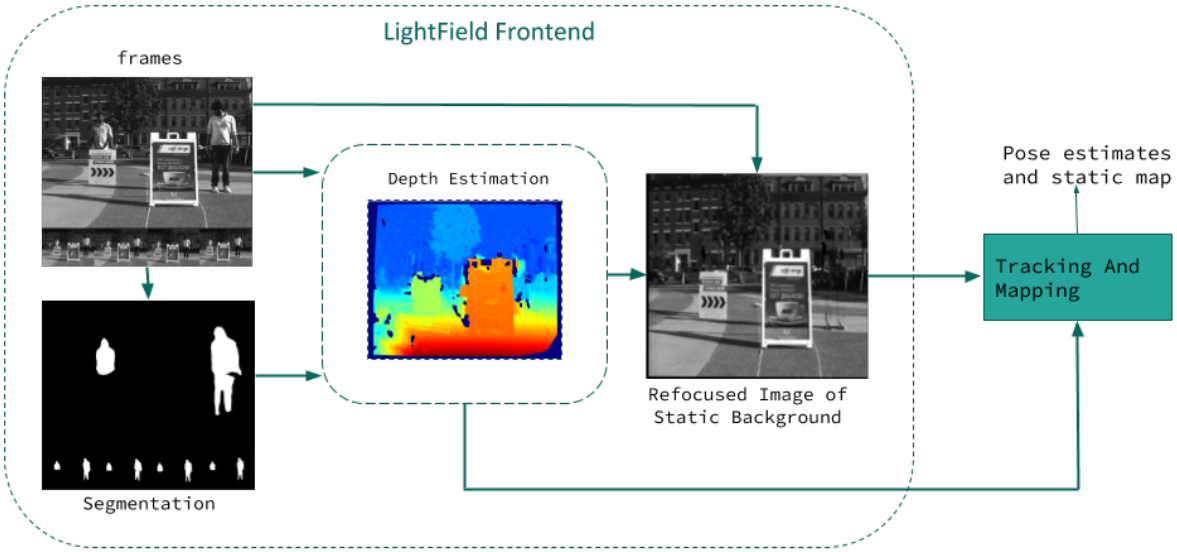


Fig. 1: Block diagram of the Light Field Front-end. Frames from the linear camera array and the segmentation masks are fed into the depth estimation module. The optimized depth is used to compute the refocused image.

other methods, we not only track features belonging to the statically segmented parts of the image but also extract and track the static features occluded by dynamic objects from the refocused image of the static background. Thus, we demonstrate increased accuracy and robustness even when the dynamic content starts to dominate the scene.

Light Fields in Robotics Light Fields are a popular topic in computer vision and graphics for refocusing and rendering[19][20][21][22], with applications that include super-resolution[23] and depth estimation[24][25][26][27][28]. Application of Light Fields in robotics is not as developed but has been discussed in[9][29][10]. Most of the robotics related work aims to solve the visual odometry problem [29], and to compensate for challenging light and weather conditions[30][10][11][31]. None of this work addresses the problem of dynamic environments. In contrast, our efforts in semantic guided refocusing and reconstruction specifically deal with dynamic objects in a SLAM framework. Refocusing[20] is performed by blending rays emerging from a focal surface and passing through a synthetic aperture without taking into account the semantics of the rays. The graphics community's Light Field depth reconstruction techniques typically are aimed at increasing the accuracy of reconstruction while trading off speed of computation. Most of the Light Field research is targeted towards micro lenslet cameras[32] which suffers from small baseline separation, which are far from ideal in terms of seeing through the dynamic objects, where wide-baseline arrays are more suitable.

III. Light Field front-end

At the heart of the Light Field front-end is our method to compute an all-in-focus synthetic aperture image of the static background from the Light Field captured with a linear camera array. The processing pipeline is illustrated in fig. 1.

A. Problem Setup

A calibrated linear array of K cameras forming images I_k can be seen as a single system with synthetic aperture where each camera contributes to a ray passing through the aperture. Following the two-plane parameterization of Light Field [20], the camera positions on the array are the (s,t) coordinates which lie on the entrance plane and the pixels are the (u,v) coordinates on the image plane. This creates a unique 4D ray (s,t,u,v) as shown in fig. 2. Given an arbitrary focal surface F and a 3D point (X,Y,Z) on it, we can combine all the rays of the synthetic aperture which emit from that point and intersect the cameras into a single refocused ray.

Suppose we have the depth map of the static background. Let d^{ref} and x^{ref} be the depth and coordinates of a pixel in the reference view, then the rays x^k in the other cameras k can be sampled using:

$$x^k = \pi_k[R_k|t_k]\pi_{ref}^{-1}(x^{ref}, d^{ref}) \tag{1}$$

where π_k is a projective mapping between the a 3D point in space and 2D pixel coordinates on the image plane, and R_k and t_k are the rotation and translation of the kth camera. These rays form our synthetic aperture and are typically combined by applying an average filter[21], [20] giving equal weight to all the rays.This helps to generate images with varying focus and depth of field [19][22]. In free-space, all the camera rays correspond to the same point on the focal surface, but, in case of occlusions, some rays get obstructed. Given a large synthetic aperture, choosing a focal surface behind the foreground object brings the background into focus and causes the object to blur. This was applied in [10] to see through rain and snow by manually selecting a focal plane on the road signs for autonomous navigation. Instead of averaging all the rays, we design a filter which selects only the rays that map to static pixels in the respective

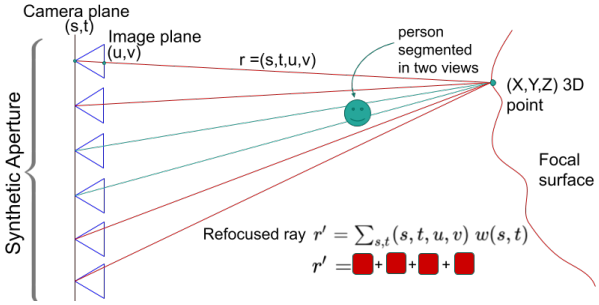


Fig. 2: Light Field refocusing. Given a 3D point on the focal surface the static rays projected into other cameras are combined to create a single refocused image.

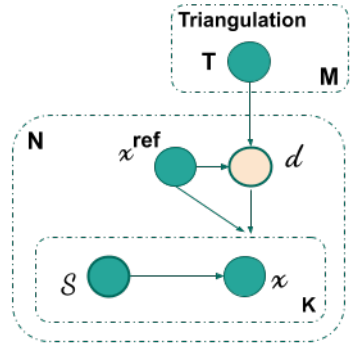


Fig. 3: The Probabilistic Graphical Model. Triangulation T and segmentation S are pre-computed. d is disparity of a pixel in the reference view, while x is a set of K reprojections into other cameras.

views obtained from semantic segmentation. This way the dynamic objects in the foreground are completely eliminated. Denoting whether a pixel x^k belongs to the static or dynamic content with s^k, the refocused image I^* can be calculated as follows:

$$I^s = \frac{\sum I(x^k) * (s^k = static)}{\sum (s^k = static)} \tag{2}$$

As we can observe in eq. (1), the depth map of the static background is required to compute the refocused image.

B. Probabilistic Graphical Model

The main challenge in the depth estimation of the static background is to reconstruct parts of the reference view occluded by dynamic objects. Given that an array of cameras are acting as the source of image data, we need to estimate the depth map D^* and refocused image I^* of static background for a reference camera view X^{ref}. The key to finding correct depth is to choose the subset of camera views which will exclude the pixels corresponding to dynamic objects while computing the image correspondences. We apply MaskRCNN[14] on camera images to compute per-pixel semantic labels. The CNN model assigns a probability to pixels being segmented $\{S_P \in \mathbb{R}^k | 0 \leq S_P^k \leq 1\}$, where S_P^k is the probability in k^{th} view. We introduce a set of binary random variables S^k assigned to each pixel i in camera k which take values either 0 or 1 representing dynamic or static pixels based on a threshold on the probability. The probabilistic graphical model is shown in the fig. 3. The disparity d can be obtained by maximizing the posterior probability $p(d|\mathbf{x}, x^{ref}, T, Sp)$ (MAP estimate) as follows:

$$d^* = argmax_d \ \ p(d|\mathbf{x}, T, S_P, x^{ref}),$$
$$p(d|\mathbf{x}, T, S_P, x^{ref}) \propto p(d, \mathbf{x}|T, S_P, x^{ref})$$

C. Disparity Estimation

According to the graphical model, the joint probability factorizes into disparity prior and pixel likelihood. Now, disparity estimation is a task of picking a value that optimizes:

$$\arg\max_d \ \log \underbrace{p(\mathbf{x}|d, x^{ref}, S)}_{\text{pixel likelihood}} + \log \underbrace{p(d|T, x^{ref})}_{\text{disparity prior}} \tag{3}$$

1) Disparity Prior: We use piece-wise planar prior over the disparity space $p(d|T, x^{ref})$ by forming triangulation on a sparse set of points similar to ELAS [33]. This helps with poorly textured regions and gives a coarse disparity map, thus reducing the search space during optimization.

First a sparse set of points are selected on a regular grid in the reference image and their descriptor vectors are created from the responses of a 5x5 window sobel filter. Since some of these points lie on the dynamic objects we use segmentation mask to filter them. The remaining points are matched along the full range of disparities on the epipolar lines of its neighbouring image. These points together with their matched disparities form the static support points. We now exploit the Light Field data to extract support points of the static background occluded by the dynamic objects in reference view, but are visible in other views. For each image other than the reference view we extract the support points by matching the sobel descriptors of the static points against its neighbouring view and choose the points that reproject onto the parts of the reference image dynamic objects. We then filter out duplicates and inconsistent support points based on their disparity values. The final set of support points are used as vertices for the delaunay triangulation. The prior on disparity $p(d|T, x^{ref})$ is modeled as combination of a uniform distribution and a sampled Gaussian centered about the interpolated disparity from the triangulation, for support points in the neighbourhood[33]. All the above steps are shown in fig. 4 (a-e).

2) Pixel Likelihood: The triangulation prior provides a coarse map of the interpolated disparities and a set of candidate disparities. which we have used to generate accurate estimates based on the pixel likelihood. Given the coordinate x^{ref} in the reference frame and a candidate disparity d, the corresponding coordinates of the source images x^k can be determined using a warping function $\mathcal{W}_k(x^{ref}, d)$. This warping function represents a homography which can be computed from the reference and k^{th} camera matrices [34]. This allows us to use generalized disparity space suitable for multi-view camera configuration. We work in the disparity

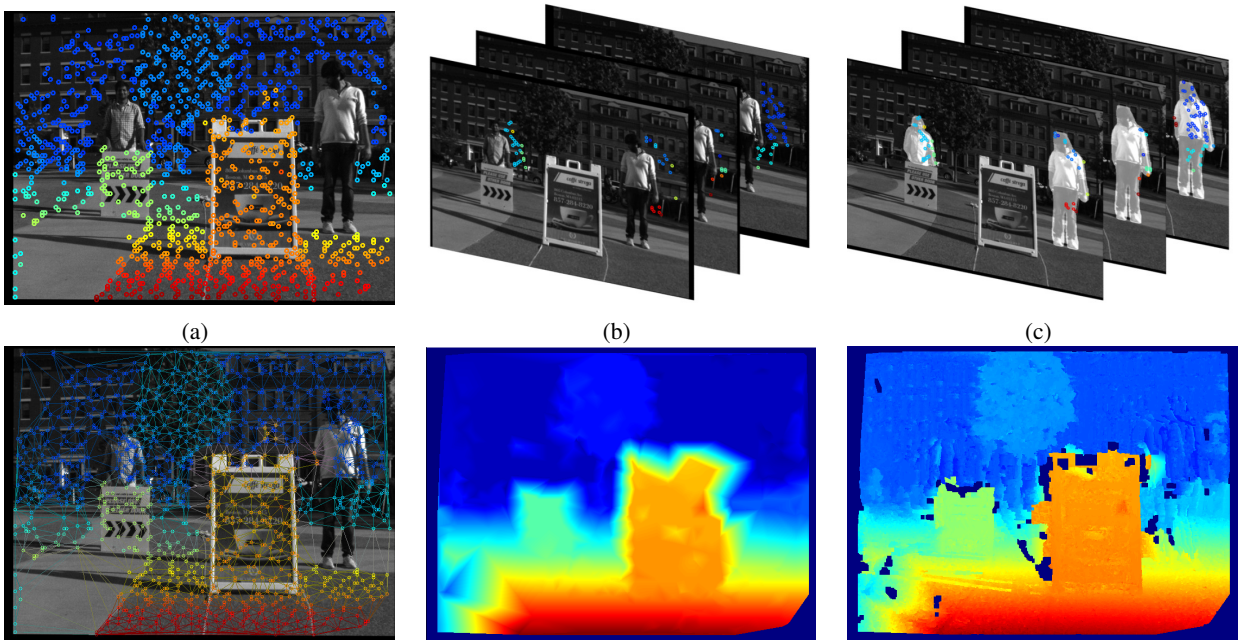


Fig. 4: Various steps associated with the static background disparity estimation. (a) A sparse set of support points only on the static background in the reference view. (b) Support points from other views. (c) Reprojected support points in the reference frame. The support points are colored based on their disparities (red: close, blue : Far). (d) Final triangulation of the static background. (e) The coarse disparity map formed by the piece-wise planar prior. (f) The final refined disparity map of the static background.

space as opposed to depth space because it facilitates discrete optimization that results in faster computation. The pixel likelihood relies on the fact that for correct disparity value there is a high probability that warped static pixels x^k in the source images will have photometric consistency. So, we design our likelihood function as a Laplace distribution restricted to static pixels such that the variance between them is minimum.

$$p(x^1...x^K \mid d, x^{ref}, S) \propto \begin{cases} exp\big(-\beta\, Var(f(x^1), ..., f(x^K))\big) \\ \quad for \;\; x^k = \mathcal{W}_k(x^{ref}, d) \\ 0 \quad otherwise \end{cases}$$

Where the variance is computed on the feature descriptors $f(x^K)$ of the pixels that are classified as static:

$$Var(...) = \frac{\sum_{k=1}^{K}(S^k = 1)[f(x^k) - \hat{f}]^2}{\sum(S^k = 1)}$$

with $\hat{f} = \frac{\sum_k (S^k=1) f(x^k)}{\sum_k (S^k=1)}$ as the mean of the descriptors. We use the descriptors created from 3x3 sobel filter responses similar to ELAS[33]. The dynamic pixels don't contribute to the likelihood as they can have arbitrary intensities. Plugging in the disparity prior and pixel likelihood into eq. (3) and taking negative logarithm gives the energy function to be minimized

D. Refocused Image Synthesis

Once the optimization converges we perform gap interpolation and median filter on the estimated disparity image to get rid of any speckle noise obtain a smooth map. The filtered disparity and segmentation masks are used to compute the refocused image of the static background using eq. (2).

IV. Experimental Analysis

In this section, we evaluate the Light Field front-end on real-world dynamic sequences.

Dataset Acquisition and Setup Light Field acquisition is typically done via a large array of cameras [19] or a micro lenslet camera [32]. The former is impractical to mount on most mobile robots where as the latter suffers from limited parallalax due to small baseline separation. Based on these constraints we use a custom built 5-camera linear array and mount it on a Clearpath Robotics Husky UGV running ROS for collecting Light Field data. The array uses 1.6MP Pointgrey BlackflyS global shutter cameras which are hardware synced for synchronized image capture and calibrated using the Kalibr[35] package. All our experiments were conducted on an Alienware gaming laptop with a 8GB GEForce GTX 1080 GPU and Intel core i7 processor. We collected multiple indoor datasets with people moving in the scene to depict dynamic environments. The ground truth trajectories were obtained using optitrack prime x13 cameras [36] which are accurate to a millimeter.

Qualitative results We show the results of our refocused image synthesis on a few Light Field snapshots from our datasets in fig. 5. The top row shows the original images and the bottom row features the corresponding refocused

Fig. 5: *Top row:* Original images in the reference view. *Bottom row:* Refocused image of the static background. First column shows the result with some low textured regions. The second column shows the same in the presence of a lot of detail. Third column shows the case when dynamic objects dominate the scene.

images. As we can see in fig. 5:*right*, our algorithm does a good job of reconstructing a complex scene when the people are obstructing our view of the majority of the background. The left image in fig. 5 presents a case of low textured regions (blue curtain and floor) and shows that the algorithm can handle such a situation quite well due to the piece-wise planar prior. In the center indoor scene we can observe the quality of the reconstruction in the presence of significant detail. In the reconstructions there are some portions of the image which are not reconstructed though it is belongsto a dynamic object eg: some portions of upper body in fig. 5:*right*. This happens when there is not enough paralallax and rays cannot reach the background and is a function of baseline separation. However, we can recover the majority of the static scene as seen in the images.

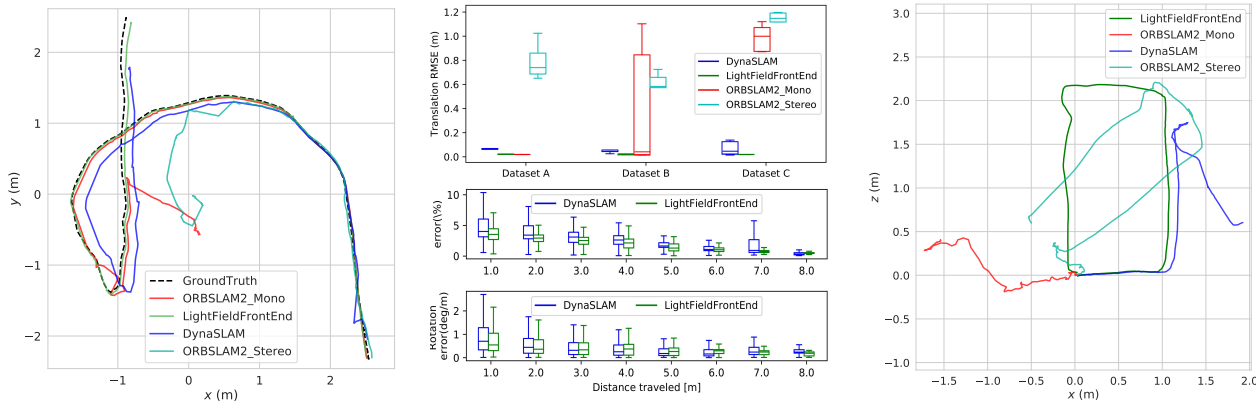
Quantitative Analysis The Light Field front-end is compared with other state of the art SLAM algorithms viz., ORBSLAM2[3] for baseline comparison and DynaSLAM[7] as it also performs static image synthesis in dynamic environments via inpainting. However, the inpainted images are not used for tracking. We demonstrate the effectiveness of our method by comparing the accuracy of pose estimation using Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) metrics. We run the algorithms 5 times on each dataset and take the median errors to account for the non-deterministic nature of system as suggested in [3]and [7]. The evaluation is conducted on three indoor sequences collected with less, moderate and significant amounts of dynamic objects by roughly moving along the same path in each sequence.

The ATE and RPE for translational and rotational components for the three cases are computed using the RPG trajectory evaluation package[37] and are shown in table I. In fig. 6 (a) the estimated trajectories are aligned and scaled with respect to the ground truth and overlaid on a plot. We can see that Light Field front-end estimates the trajectory closest to the ground truth amongst other algorithms. We observe that ORBSLAM2 Monocular version shows slightly better accuracy than the Light Field front-end in the sequence with less dynamic content due to its robust initialization and outlier rejection strategy. Even though the initialization phase helps with accurate tracking it also often delays bootstrapping in the presence of dynamic objects. This results in missing parts of the estimated trajectory. The performance deteriorates as the proportion of dynamic content increases because the outlier rejection is no longer effective. ORBSLAM2 stereo shows highest errors across all the datasets because the system is initialized from the first frame causing the dynamic keypoints to be tracked.

Both DynaSLAM and Light Field front-end outperform ORBSLAM2(Mono and Stereo) as the dynamic content increases. Our Light Field front-end achieves better accuracy than DynaSLAM in all the datasets except for the relative rotational error (RRE) in dataset b. However we achieve a lower overall RTE and RRE per meter computed for all the datasets when compared to DynaSLAM as shown in fig. 6 (b). While DynaSLAM maintains consistent ATE and RTE values across the three datasets, we can see from fig. 6 (b) that the spread of ATE increases with the dynamic content. This is because in DynaSLAM the features corresponding to the dynamic objects are discarded and only the remaining features are used in the low-cost tracking module for pose estimation. This can introduce errors in tracking when the dynamic features dominate the scene. The Light Field front-end on the other hand demonstrates consistent performance across datasets because instead of discarding dynamic key

Algorithm	Less Dynamics (A)			Moderate Dynamic (B)			More Dynamics (C)		
	ATE[m] mean , median	RTE [%]	RRE [°/m]	ATE[m] mean , median	RTE [%]	RRE [°/m]	ATE[m] mean , median	RTE [%]	RRE [°/m]
ORBSLAM2 Mono	**0.021 , 0.019**	**3.474**	**0.404**	0.399 , 0.019	50.987	1.238	0.987 , 1.000	138.24	10.798
ORBSLAM2 Stereo	0.792 , 0.739	106.917	8.624	0.563 , 0.584	151.94	10.51	1.028 , 1.148	126.78	9.057
DynaSLAM	0.068 , 0.067	5.671	0.724	0.057 , 0.044	6.22	**0.494**	0.065 , 0.046	5.126	0.846
Light Field front-end	0.022 , 0.021	3.492	0.439	**0.023 , 0.018**	**3.66**	0.57	**0.021 , 0.020**	**3.943**	**0.733**

TABLE I: Comparison of the ATE [m], RTE(%) and RRE(°/m) of various visual SLAM algorithms with Light Field front-end in dynamic scenes. The lowest error values are shown in bold font for convenience



(a) (b) (c)

Fig. 6: (a)Example trajectory plot for ORBSLAM2 Mono, Stereo, DynaSLAM and Light Field front-end compared against the ground truth obtained from optitrack mocap system. (b) *top:* The Absolute Trajectory Error (ATE) for the datasets A,B,C.*center:* The overall RTE (%) and *bottom:* RRE (°/m) for different sub trajectory lengths. (c) Trajectories of ORBSLAM2 Mono, Stereo and DynaSLAM compared with that of Light Field front-end for the dataset with loop closure.

points associated with segmented pixels we recover the static background, thus gaining the support of static features.

We further prove that our method is effective in loop closures in the presence of dynamic objects as shown in the fig. 6(d). This sequence is a rectangular path recorded with people present in the view at the start of the loop. These people, ”dyamic objects”, move as the trajectory progresses and finally go out of view when we come back to finish the loop. Dynamic objects introduce errors in pose estimation at different points along the trajectory. Monocular ORBSLAM2 experiences extreme errors due to bad initialization. Stereo ORBSLAM2 shows errors initially and DynaSLAM loses tracking at the middle of the trajectory; Both of them recover at a later point but none of the other algorithms achieve loop closure when we come back to the starting point because the feature distribution has changed. Finally, we report the

Dataset	depth map [ms]	refocusing[ms]	Total Avg Time[ms]
Dataset A	153.354	13.297	208.717
Dataset B	155.482	14.357	233.918
Dataset C	159.569	14.518	246.831

TABLE II: Average time taken by various processing steps.

average computational time for carrying out the different stages of the front-end processing for various sequences in table II. To speed up processing we refocus only the dynamic pixels as the other pixels already have the static background intensities. Thus, the computational complexity scales with respect to the amount of dynamic content. Due to the GPU implementation we can achieve speed of 4.35 fps facilitating close to real-time performance.

V. Conclusion

We presented a Light Field SLAM front-end where we compute the depth map and synthetic aperture image of the static background using semantic segmentation of apriori dynamic objects. The synthesized image allows us to extract static features occluded by dynamic objects resulting in better tracking accuracy and robustness. This approach performs well even when the scene is dominated by dynamic objects, eliminates the need for an initialization phase, and enables parallel processing resulting in close to real-time executions at 3 fps on a laptop with a GPU. We show promising results by evaluating our algorithm on real-world datasets collected with dynamic content. In the future, we would like to tightly couple the apriori semantic knowledge with the dynamics of the scene to detect actively moving objects and develop a pose graph back-end for dynamic Light Field SLAM.

Acknowledgments

This work was funded in part by ONR Grant N00014-19-1-2131. We would like to thank S Ramalingam, A Techet and A Bajpayee for their guidance and NU Field Robotics lab for helping with data collection.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual slam and structure from motion in dynamic environments: A survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 2, pp. 1–36, 2018.

[3] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[4] W. Tan, H. Liu, Z. Dong, G. Zhang, and H. Bao, "Robust monocular slam in dynamic environments," in *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2013, pp. 209–218.

[5] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.

[6] P. F. Alcantarilla, J. J. Yebes, J. Almazán, and L. M. Bergasa, "On combining visual slam and dense scene flow to increase the robustness of localization and mapping in dynamic environments," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 1290–1297.

[7] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "Dynaslam: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4076–4083, 2018.

[8] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Ds-slam: A semantic visual slam towards dynamic environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1168–1174.

[9] F. Dong, S.-H. Ieng, X. Savatier, R. Etienne-Cummings, and R. Benosman, "Plenoptic cameras in real-time robotics," *The International Journal of Robotics Research*, vol. 32, no. 2, pp. 206–217, 2013.

[10] A. Bajpayee, A. H. Techet, and H. Singh, "Real-time light field processing for autonomous robotics," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4218–4225.

[11] N. Zeller, F. Quint, and U. Stilla, "Scale-awareness of light field camera based visual odometry," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 715–730.

[12] R. Scona, M. Jaimez, Y. R. Petillot, M. Fallon, and D. Cremers, "Staticfusion: Background reconstruction for dense rgb-d slam in dynamic environments," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–9.

[13] D.-H. Kim and J.-H. Kim, "Effective background model-based rgb-d dense visual odometry in a dynamic environment," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1565–1573, 2016.

[14] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[15] T. L. Zhu and D. Oved, "BodyPix - Person Segmentation in the Browser," 2019.

[16] I. A. Bârsan, P. Liu, M. Pollefeys, and A. Geiger, "Robust dense mapping for large-scale dynamic environments," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7510–7517.

[17] E. Palazzolo, J. Behley, P. Lottes, P. Giguere, and C. Stachniss, "Refusion: 3d reconstruction in dynamic environments for rgb-d cameras exploiting residuals," *arXiv preprint arXiv:1905.02082*, 2019.

[18] J. Zhang, M. Henein, R. Mahony, and V. Ila, "VDO-SLAM: A Visual Dynamic Object-aware SLAM System," 2020.

[19] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996, pp. 31–42.

[20] A. Isaksen, L. McMillan, and S. J. Gortler, "Dynamically reparameterized light fields," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 297–306.

[21] T. G. Georgiev and A. Lumsdaine, "Focused plenoptic camera and rendering," *Journal of electronic imaging*, vol. 19, no. 2, p. 021106, 2010.

[22] A. Davis, M. Levoy, and F. Durand, "Unstructured light fields," in *Computer Graphics Forum*, vol. 31, no. 2pt1. Wiley Online Library, 2012, pp. 305–314.

[23] T. E. Bishop and P. Favaro, "The light field camera: Extended depth of field, aliasing, and superresolution," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 5, pp. 972–986, 2011.

[24] V. Vaish, M. Levoy, R. Szeliski, C. L. Zitnick, and S. B. Kang, "Reconstructing occluded surfaces using synthetic apertures: Stereo, focus and robust measures," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 2331–2338.

[25] S. Wanner and B. Goldluecke, "Globally consistent depth labeling of 4d light fields," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 41–48.

[26] T.-C. Wang, A. A. Efros, and R. Ramamoorthi, "Occlusion-aware depth estimation using light-field cameras," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3487–3495.

[27] I. K. Park, K. M. Lee, *et al.*, "Robust light field depth estimation using occlusion-noise aware data costs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 10, pp. 2484–2497, 2017.

[28] H. Sheng, P. Zhao, S. Zhang, J. Zhang, and D. Yang, "Occlusion-aware depth estimation for light field using multi-orientation epis," *Pattern Recognition*, vol. 74, pp. 587–599, 2018.

[29] D. G. Dansereau, I. Mahon, O. Pizarro, and S. B. Williams, "Plenoptic flow: Closed-form visual odometry for light field cameras," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 4455–4462.

[30] K. A. Skinner and M. Johnson-Roberson, "Towards real-time underwater 3d reconstruction with plenoptic cameras," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2014–2021.

[31] J. Eisele, Z. Song, K. Nelson, and K. Mohseni, "Visual-inertial guidance with a plenoptic camera for autonomous underwater vehicles," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2777–2784, 2019.

[32] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, P. Hanrahan, *et al.*, "Light field photography with a hand-held plenoptic camera," *Computer Science Technical Report CSTR*, vol. 2, no. 11, pp. 1–11, 2005.

[33] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Asian conference on computer vision*. Springer, 2010, pp. 25–38.

[34] R. Szeliski and P. Golland, "Stereo matching with transparency and matting," *International Journal of Computer Vision*, vol. 32, no. 1, pp. 45–61, 1999.

[35] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1280–1286.

[36] "Prime 13 - in depth." [Online]. Available: https://optitrack.com/products/primex-13/

[37] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7244–7251.

Manuscript 1563 submitted to 2021 IEEE International Conference on Robotics and Automation (ICRA). Received October 31, 2020.