LOW POWER ADAPTIVE TIME-OF-FLIGHT IMAGING FOR MULTIPLE RIGID OBJECTS

James Noraky[1], *Charles Mathy*[2], *Alan Cheng*[1], *Vivienne Sze*[1]

[1] Massachusetts Institute of Technology, [2] Analog Devices

ABSTRACT

Time-of-flight (TOF) cameras are becoming increasingly popular for many mobile applications. To obtain accurate depth maps, TOF cameras must emit many pulses of light, which consumes a lot of power and lowers the battery life of mobile devices. However, lowering the number of emitted pulses results in noisy depth maps. To obtain accurate depth maps while reducing the overall number of emitted pulses, we propose an algorithm that adaptively varies the number of pulses to infrequently obtain high power depth maps and uses them to help estimate subsequent low power ones. To estimate these depth maps, our technique uses the previous frame by accounting for the 3D motion in the scene. We assume that the scene contains independently moving rigid objects and show that we can efficiently estimate the motions using just the data from a TOF camera. The resulting algorithm estimates 640×480 depth maps at 30 frames per second on an embedded processor. We evaluate our approach on data collected with a pulsed TOF camera and show that we can reduce the mean relative error of the low power depth maps by up to 64% and the number of emitted pulses by up to 81%.

***Index Terms*—** time-of-flight cameras, shot noise, depth estimation, motion estimation, low power

1. INTRODUCTION

Time-of-flight (TOF) cameras are becoming increasingly popular for mobile applications. These depth sensors are appealing because they obtain dense depth maps with minimal latency [1]. For mobile devices, one drawback of a TOF camera is that its illumination source can be power hungry. This is apparent for an application like augmented reality (AR), which requires the TOF camera to emit many pulses of light to obtain accurate depth maps in real time (30 frames per second) and limits the battery life of the underlying device. While reducing the number of pulses would extend the battery life, it would also result in inaccurate depth maps.

To lower the power required to estimate accurate depth maps, we propose an intermediate approach, depicted in Figure 1, where we adaptively vary the number of pulses to infrequently obtain high power depth maps and use them to help estimate subsequent low power ones. Our work is similar in spirit to previous approaches [2, 3, 4, 5], but differs in that these approaches are focused on reducing the noise for a *fixed* pulse count and adapt different settings such as pulse timing.

We thank Analog Devices for funding this work.

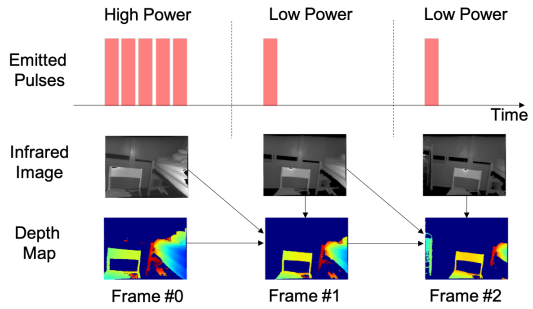

Fig. 1: **Adaptive Depth Estimation**: We adaptively vary the number of pulses a TOF camera emits. For low power depth maps, our depth estimation algorithm uses data from the previous frame along with the current one.

In order to use the data from the previous high power depth map, our algorithm must account for the 3D motion between frames. This is difficult because this motion needs to be estimated for every pixel in real time. One common way to reduce this complexity is to assume that the environment around the TOF camera is rigid [6]. We further relax this and allow independently moving rigid objects. Our assumption allows us to efficiently estimate the 3D motion using the optical flow, or the apparent pixel-wise motion, of consecutive infrared (IR) images that a TOF camera simultaneously obtains in the process of estimating depth. Unlike our previous work [7, 8], which required RGB images to estimate motion, this approach uses only the data from a TOF camera. We present the following contributions:

- An algorithm that estimates the motion of independent rigid objects using just the data from a TOF camera.
- A depth estimation algorithm that combines the previous depth map with the current low power one, decreasing its mean relative error by up to 64%.
- A method to determine when a high power depth map should be obtained, reducing the number of emitted pulses by up to 81%.

The resulting algorithm is computationally efficient, and we highlight our design choices that enable our approach to es-

timate depth maps at 30 frames per second on the ODROID XU-3 [9] embedded platform.

2. TOF CAMERA NOISE ANALYSIS

To quantify the impact of power on the quality of the depth map estimated by a TOF camera, we first examine how a pulsed TOF camera estimates depth. These sensors obtain depth by emitting light and estimating its roundtrip time. To estimate the roundtrip time, the reflected light is integrated by the sensor, where the light is first converted into electrons and then quantized by an analog-to-digital converter.

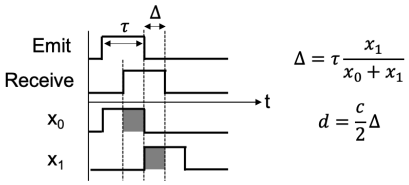

Fig. 2: **Pulsed TOF Camera**: x_0 and x_1 correspond to the shaded regions, which represent the integrated light.

In Figure 2, we show how the roundtrip time, Δ, is estimated for a single pulse. In this figure, we denote d as the depth, x_0 and x_1 as the integrated "light", τ as the pulse width, and c as the speed of light. We assume that any contribution from the background is subtracted from x_0 and x_1. This process is repeated for every pixel, and in addition to obtaining a depth map (d), we also obtain an IR image ($x_0 + x_1$).

However, x_0 and x_1 are affected by Poisson noise, or shot noise, due to the discrete nature of electrons. Shot noise is always present in TOF cameras, and we also assume that it is the dominant noise source [10]. We can then model x_0 and x_1 as scaled and independent Poisson random variables, i.e. x_0/α and x_1/α are Poisson distributed for some constant α. With this assumption, we can derive the depth variance, σ_d^2, by applying error propagation to d:

$$\sigma_d^2 = d\left(\frac{c\tau}{2} - d\right)\frac{\alpha}{x_0 + x_1} \quad (1)$$

From Eq. (1), we see that σ_d^2 is inversely proportional to the intensity of the reflected light, or $x_0 + x_1$. This inverse relationship means that the variance is *especially high* for low intensities and highlights the challenges of obtaining accurate depth maps with low power. One way to increase the intensity is to increase the number of emitted pulses, but as we previously mentioned, this would limit battery life.

3. LOW POWER DEPTH ESTIMATION

To estimate low power depth maps, our algorithm (Figure 3) uses the previous depth map and IR image pair as well as the current one. Our approach estimates the motion between the frames and uses it to obtain the final depth map.

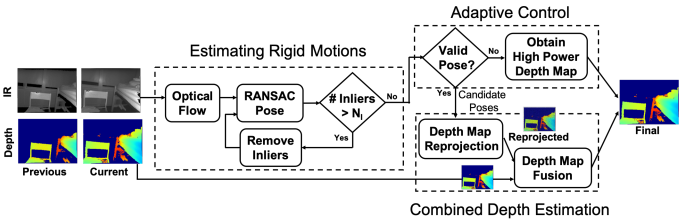

Fig. 3: **Algorithm Pipeline**: We apply our algorithm to estimate low power depth maps.

3.1. Estimating Rigid Motions

Our approach assumes that the scene contains independently moving rigid objects. One conceptually straightforward way to estimate these motions is to segment the IR image into rigid regions and estimate the motion for each region. However, many segmentation algorithms are computationally expensive [11], and thus, we propose an alternative approach.

If a set of n pixels in the previous depth map, $\mathcal{P} = \{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \Omega$, have the same rigid motion, represented by the pose composed of a rotation, R, and a translation, T, then the following holds for each pixel:

$$\left\| \left(R\frac{z_i}{f}\begin{bmatrix} x_i - x_c \\ y_i - y_c \\ f \end{bmatrix} + T \right) \times \begin{bmatrix} x_i + u_i - x_c \\ y_i + v_i - y_c \\ f \end{bmatrix} \right\|_2^2 = 0 \quad (2)$$

where z_i and (u_i, v_i) are the depth and optical flow of the i^{th} pixel and f and (x_c, y_c) are the focal length and the principal point of the TOF camera. In practice, Eq. (2) is not satisfied exactly, but we can still use it to group pixels that have the same rigid motion. To do so, we extend our previous approach [7] by modifying RANSAC to estimate the pose with the *largest inlier set*. We assume that this pose corresponds to the motion of the largest rigid region in the image. We repeat this process after removing the pixels of the inlier set, i.e. $\Omega \setminus \mathcal{P}$, to find the pose for the next largest rigid region and continue until the size of the inlier set falls below a threshold (N_i). By following this procedure, we estimate the independent rigid motions in the scene.

We also further reduce computation by only considering the pixels on a sparse and uniformly spaced grid. This minimizes computation because we do not need to localize keypoints or compute a dense optical flow field. In our implementation, we estimate optical flow using block matching with normalized correlation. We perform block matching using 15×15 blocks centered on the pixels of a 20×20 grid and search a 45×45 region. We also set $N_i = 20$.

3.2. Combined Depth Estimation

In the previous section, we obtain the pose of the independently moving rigid objects in the scene. Here, we describe how we use the estimated pose to obtain the final depth map.

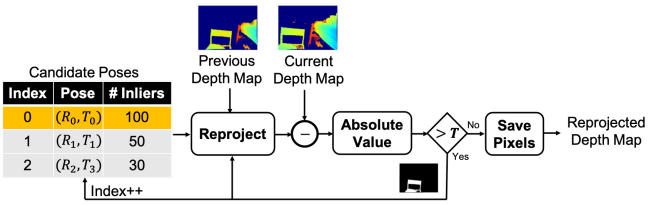

Fig. 4: **Depth Map Reprojection Pipeline**: We apply the poses in order of its inlier set size and use the current depth map to help assign the correct pose to each pixel.

3.2.1. Depth Map Reprojection

We update the previous depth map by reprojecting it, where we first obtain the new 3D position of each pixel using the estimated pose and then project its updated depth to an image. Because we can have many candidate poses, we need to assign the right one to each pixel. One straightforward approach is to reproject the previous depth map using each pose, which results in many candidate depth maps. To determine the final reprojected depth map, for each pixel, we choose the candidate depth that is closest to that in the current depth map. However, this adds a significant overhead because we need to reproject the *entire depth map* for each pose.

We take a different approach as shown in Figure 4. We first use the pose that has the largest inlier set to reproject the previous depth map. As described in the previous section, we assume that this pose corresponds to the largest rigid region in the image. To determine the pixels of this region, we compare the reprojection to the current depth map and apply a threshold (T) to the absolute difference. We save the reprojected pixels whose difference are within the threshold and repeat this process with the remaining pixels using the poses in order of the inlier set sizes. Because we proceed in this order, we reduce the total computation for reprojection. In Section 4.2, we show that this pose assignment is consistent with the underlying rigid motion.

3.2.2. Depth Map Fusion

To obtain the final depth map, we combine the reprojected depth map with the current one using a weighted average. It can be shown that choosing the weights based on the variances of the depth lowers the expected mean squared error. Denoting $\sigma^2_{i,t}$ as the variance of the i^{th} pixel in the current depth map and $\sigma^2_{i,t-1}$ as that in the reprojected one, we obtain the final depth as follows:

$$\hat{z}_{i,t} = \frac{\sigma^2_{i,t}}{\sigma^2_{i,t} + \sigma^2_{i,t-1}} z_{i,t-1} + \frac{\sigma^2_{i,t-1}}{\sigma^2_{i,t} + \sigma^2_{i,t-1}} z_{i,t} \qquad (3)$$

where $\hat{z}_{i,t}$ is the final depth for the i^{th} pixel, $z_{i,t-1}$ is the reprojected depth, and $z_{i,t}$ is the current one.

These variances must be computed for *every pixel* using Eq. (1) and also updated as it is reprojected. To reduce the computation, we approximate the weights as follows:

$$\frac{\sigma^2_{i,t-1}}{\sigma^2_{i,t-1} + \sigma^2_{i,t}} \approx \frac{1}{M+1} \qquad (4)$$

where M is the ratio of the number of emitted pulses for a high power depth map to that of a low power one. This approximation shows that more weight is given to the reprojected depth, which makes sense because it contains information from the last high power depth map. Furthermore, in our experiments, we find that this approximation had a negligible impact on the mean relative error (as defined in Section 4.1) of our final depth maps.

3.3. Adaptive Control

As shown in Figure 3, we obtain a high power depth map when the motion cannot be estimated. This occurs when accurate optical flow cannot be obtained due to issues like the lack of features in the IR images.

4. ALGORITHM EVALUATION

4.1. Results

Datasets To evaluate our algorithm, we capture 640×480 IR image and depth map pairs using the Pico Zense TOF camera [12]. In our experiments, the high power depth maps are obtained with $10\times$ as many pulses as the low power ones. We obtain two datasets shown in Figure 5:

1. *Indoor Sequences*: This synthetic dataset is obtained by moving the TOF camera around different indoor environments. We use the captured data as ground truth and simulate high and low power depth maps by adding shot noise to the measurements.

2. *Rail Sequence*: This dataset is collected using a rail that moves a calibration target away from the TOF camera. We move the target in 5 cm increments and capture 300 depth maps. We use a single depth map to represent a low power depth map, average 10 for a high power one, and average all 300 for the ground truth.

The rail sequence also tests the performance of our algorithm for scenes with multiple rigid objects since the calibration target moves while its surroundings does not. We examine other cases of multiple rigid objects in Section 4.2.

Fig. 5: **Datasets**: Select indoor and rail sequences shown.

Implementation We implement our depth estimation algorithm on the ODROID XU-3 board [9], which is an embedded platform with an Exynos 5422 processor (used in Samsung Galaxy S5 [13]). It outputs depth maps in real time for the data we evaluate on, and the processing time is spent equally between estimating the rigid motions and the combined depth estimation.

Methodology We apply our algorithm to our sequences and use the high power depth maps when it is required. For each depth map, we compute the percent mean relative error (MRE) over the N pixels we estimate. This is equal to $\frac{100}{N}\sum_{i=1}^{N}\frac{|\hat{z}_i - z_i|}{z_i}$, where z_i and $\hat{z}_i$ are the ground truth and estimated depth, respectively. We also note the percentage of high power depth maps we obtain.

We summarize the performance of our algorithm in Table 1, where we average the MRE across all of the depth maps. We see that the MRE of our depth maps are 64% and 63% lower than that of the low power ones for the indoor and rail sequences, respectively. Our algorithm also only obtains high power depth maps for 10% and 18% of these frames for the respective sequences. This means that we reduce the overall number of emitted pulses by up to 81%. An example of an estimated depth map (This Work) is shown in Figure 6, where it is compared against the low (Low Power) and high (High Power) power depth maps. We see that our depth map is less noisy and visually sharper than the low power one.

Configuration	Indoor	Rail
This Work	3.2%	4.3%
High Power	2.6%	3.7%
Low Power	8.8%	11.5%
This Work+Bilateral Filter	2.3%	3.4%
Low Power+Bilateral Filter	6.3%	11.0%
Equivalent Power	6.2%	6.8%

Table 1: **Mean Relative Error**: Our adaptive algorithm only obtains high power depth maps for 10% and 18% of the frames for the indoor and rail sequences, respectively.

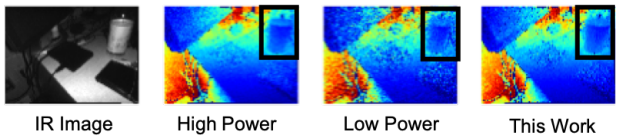

Fig. 6: **Example Depth Maps**: Our estimated depth map is less noisy than the low power one.

4.2. Discussion

Comparison To Denoising Methods Even though our approach is focused on estimating accurate depth maps, we compare our depth maps to denoised low power ones. A common denoising filter is the bilateral filter [14], and it has been shown to give competitive results [15]. To compare our approach, we apply a 5×5 bilateral filter to the low power depth maps and compute its MRE (Low Power+Bilateral Filter). As shown in Table 1, our techniques still estimates depth with a lower MRE. We can also apply the bilateral filter to our depth maps (This Work+Bilateral Filter), which further reduces the MRE.

Comparison to Equivalent Power Because our algorithm adaptively switches between obtaining high power depth maps and low power ones, we also compare our depth maps to those obtained with a constant pulse count of equivalent power. We can simulate these depth maps by adding the appropriate noise and combining the proper number of frames for the indoor and rail sequences. If these depth maps have a lower MRE, then using the equivalent pulse count is more desirable because standard depth estimation is computationally simple. As shown in the Table 1, this (Equivalent Power) is not the case, and our approach has a lower MRE.

Estimating Multiple Rigid Motions Another feature of our approach that distinguishes us from [7, 8] is that we can estimate independent rigid motions in the scene *without* prior segmentation. We illustrate this in Figure 7, where we apply our algorithm to scenes that have multiple rigid objects. In the third column of this figure, we color code each pixel according to its assigned pose as described in Section 3.2.1. We see that the pose assignment is consistent with the rigid objects that have different motions. This qualitatively shows that our approach estimates and assigns the correct motion.

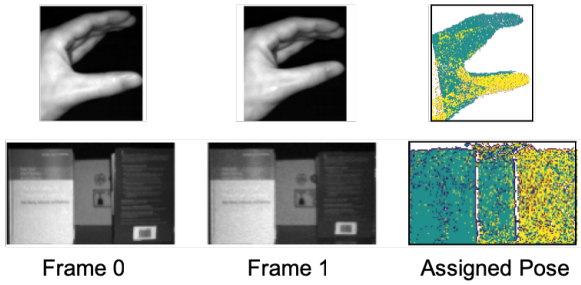

Fig. 7: **Assigning Pose**: Examples showing that the assigned pose is consistent with the rigid object motion in the scene.

5. CONCLUSION

In this work, we propose a technique to lower the total number of pulses a TOF camera emits to obtain accurate depth maps. Our algorithm adaptively varies the number of pulses to infrequently obtain high power depth maps and uses them to enhance low power ones. To do so, we efficiently estimate the 3D motion between frames using only the data from a TOF camera. Our algorithm estimates depth maps in real time on an embedded processor and reduces the MRE of the low power depth maps by 64% and the overall number of pulses by 81%.

6. REFERENCES

[1] Miles Hansard, Seungkyu Lee, Ouk Choi, and Radu Horaud, *Time-of-Flight Cameras*, SpringerBriefs in Computer Science. Springer London, London, 2013.

[2] Stefan May, Bjorn Werner, Hartmut Surmann, and Kai Pervolz, "3D Time-of-Flight Cameras for Mobile Robotics," in *International Conference on Intelligent Robots and Systems*. oct 2006, pp. 790–795, IEEE.

[3] Pablo Gil, Jorge Pomares, and Fernando Torres, "Analysis and Adaptation of Integration Time in PMD Camera for Visual Servoing," in *International Conference on Pattern Recognition*. aug 2010, pp. 311–315, IEEE.

[4] Thomas Hoegg, Christian Baiz, and Andreas Kolb, "Online Improvement of Time-of-Flight Camera Accuracy By Automatic Integration Time Adaption," in *International Symposium on Signal Processing and Information Technology*. dec 2015, pp. 613–618, IEEE.

[5] Michael Schober, Amit Adam, Omer Yair, Shai Mazor, and Sebastian Nowozin, "Dynamic Time-of-Flight," in *Conference on Computer Vision and Pattern Recognition*. jul 2017, pp. 170–179, IEEE.

[6] Juan-Antonio Fernández-Madrigal and José Luis Blanco Claraco, *Simultaneous Localization and Mapping for Mobile Robots*, Advances in Computational Intelligence and Robotics. IGI Global, 2013.

[7] James Noraky and Vivienne Sze, "Low Power Depth Estimation for Time-of-Flight Imaging," in *International Conference on Image Processing*. sep 2017, pp. 2114–2118, IEEE.

[8] James Noraky and Vivienne Sze, "Depth Estimation of Non-Rigid Objects for Time-Of-Flight Imaging," in *International Conference on Image Processing*. oct 2018, pp. 2925–2929, IEEE.

[9] HardKernel, "ODROID-XU3," `www.hardkernel.com/main/products/prdt_info.php?g_code=g140448267127`, Accessed: 2018-07-24.

[10] Julio Illade-Quinteiro, Víctor Brea, Paula López, Diego Cabello, and Gines Doménech-Asensi, "Distance Measurement Error in Time-of-Flight Sensors Due to Shot Noise," *Sensors*, vol. 15, no. 3, pp. 4624–4642, feb 2015.

[11] Roberto Tron and Rene Vidal, "A Benchmark for the Comparison of 3-D Motion Segmentation Algorithms," in *Conference on Computer Vision and Pattern Recognition*. jun 2007, pp. 1–8, IEEE.

[12] "Pico Zense DCAM100," `picozense.picovr.com/`, Accessed: 2018-07-24.

[13] Samsung, "Exynos 5422," `www.samsung.com/semiconductor/minisite/exynos/products/mobileprocessor/exynos-5-octa-5422/`, Accessed: 2018-07-24.

[14] Alexander Seitel, Thiago R. dos Santos, Sven Mersmann, Jochen Penne, Anja Groch, Kwong Yung, Ralf Tetzlaff, Hans-Peter Meinzer, and Lena Maier-Hein, "Adaptive Bilateral Filter for Image Denoising and Its Application to In-Vitro Time-of-Flight Data," mar 2011, p. 796423.

[15] Frank Lenzen, Kwang In Kim, Henrik Schäfer, Rahul Nair, Stephan Meister, Florian Becker, and Christoph S Garbe, "Denoising Strategies for Time-of-Flight Data," in *Time-of-Flight and Depth Imaging: Sensors, Algorithms, and Applications*, Marcin Grzegorzek, Christian Theobalt, Andreas Kolb, Christian Theobalt, and Reinhard Koch, Eds., vol. 8200, pp. 25–45. Springer, 2013.