Inertial Safety from Structured Light: Supplementary Technical Report

Sizhuo Ma and Mohit Gupta[1]

[1]University of Wisconsin-Madison, Madison WI 53706, USA
{sizhuoma,mohitg}@cs.wisc.edu

In this report, we provide technical details, analysis and experimental results that are not included in the main paper due to space constraints. Please refer to the supplementary video as well for the experimental results on video sequences.

1 Dependence of Accuracy on Scene and Imaging Parameters

In this section, we study the dependence of the accuracy of the estimated ISM on various parameters, including properties of the scene (albedo frequency, depth variation frequency, amount of depth change between frames), as well as properties of the structured light system (frequency of the sinusoid pattern, baseline). To give a quantitative comparison between different configurations, we report the mean relative error of the disparity change $\Delta\Upsilon$, which is defined as

$$e = \frac{1}{n}\sum_{i=1}^{n}\frac{\widehat{\Delta\Upsilon}_i - \Delta\Upsilon_i}{\Delta\Upsilon_i}, \tag{1}$$

where i is the pixel index, n is the total number of pixels, $\widehat{\Delta\Upsilon}_i$ is the estimated disparity change, and $\Delta\Upsilon_i$ is the ground truth. Notice that we don't report the error of the ISM directly. This is because the ISM S_E is related to disparity change $\Delta\Upsilon$ by $S_E = \frac{fb}{\Delta\Upsilon}$, where f is the focal length and b is the baseline. Since $\Delta\Upsilon$ appears in the denominator, for a small $\Delta\Upsilon$, a small error in $\Delta\Upsilon$ may result in a large error in S_E. As a result, errors at pixels with small $\Delta\Upsilon$ will skew the overal mean error, making the comparison between different scenarios difficult. Therefore, we report the relative error of $\Delta\Upsilon$ estimates. Since S_E is inversely proportional to $\Delta\Upsilon$, this error metric is also an estimate of the relative error in S_E.

1.1 Scene Albedo Frequency vs. Pattern Frequency

Recall that in Eq. 6 of the main paper, the structured light imaging model is given as

$$i(u,v,t) = \alpha(u,v,t)P(c) + \beta(u,v,t) \tag{2}$$

where (u,v,t) is the spatial-temporal pixel index, $P(c)$ is the corresponding column in the projector pattern. α is the reflectance term that captures various radiometric factors including distance fall-off, foreshortening, BRDF and scene albedo. β is the ambient term that also depends on ambient light.

The proposed ISM algorithm assumes α and β varies smoothly along each row (along the orthogonal direction of oriented filters if they are used). When this assumption is violated but the high frequency component of scene texture has relatively low strength, the method can still recover the ISM but with some artifacts. For example, in Fig. 2 of the main paper, the recovered ISM is not spatially smooth, especially near the object boundaries. When the scene texture contains high power in high frequency components, the method may fail, as shown in Fig. 1. The scene consists of a single plane with a bisinusoidal albedo. We

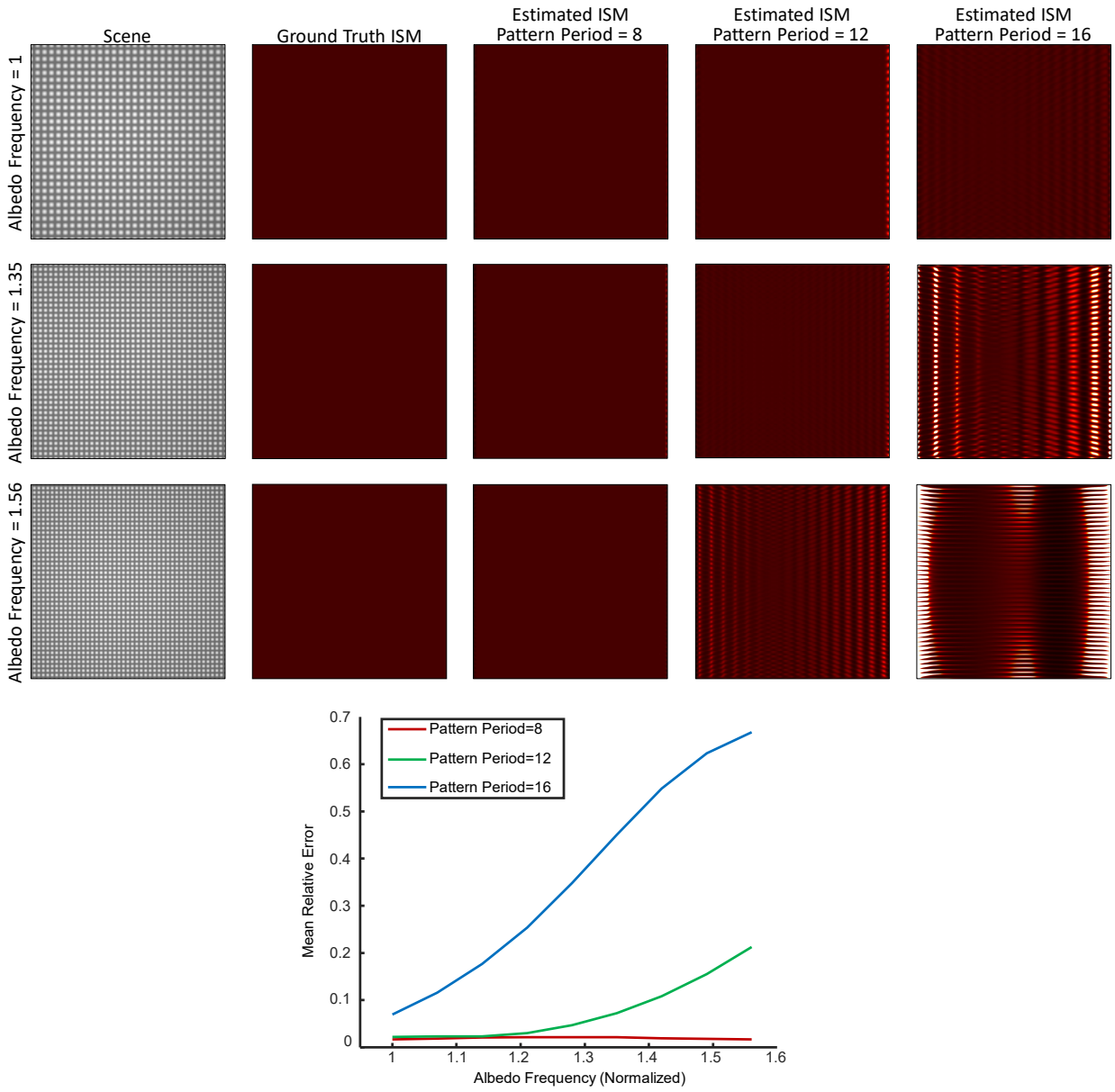

Figure 1: **Scene albedo frequency vs. pattern frequency.** The scene consists of a single plane with bisinusoidal albedo. Camera moves in z-direction. Higher albedo frequency results in higher error because the spectra of F and G cannot be completely separated. Patterns with higher frequency can better resolve this problem and give more accurate results.

plot the mean relative error for different albedo frequencies. We also compare the error for three different pattern frequencies, with periods 8, 12 and 16 respectively.

From the plot we can see that higher albedo pattern results in larger error since there is more interaction between the DC component F and the first harmonic G in the frequency domain. Patterns with higher frequency are able to separate them better and therefore get better results.

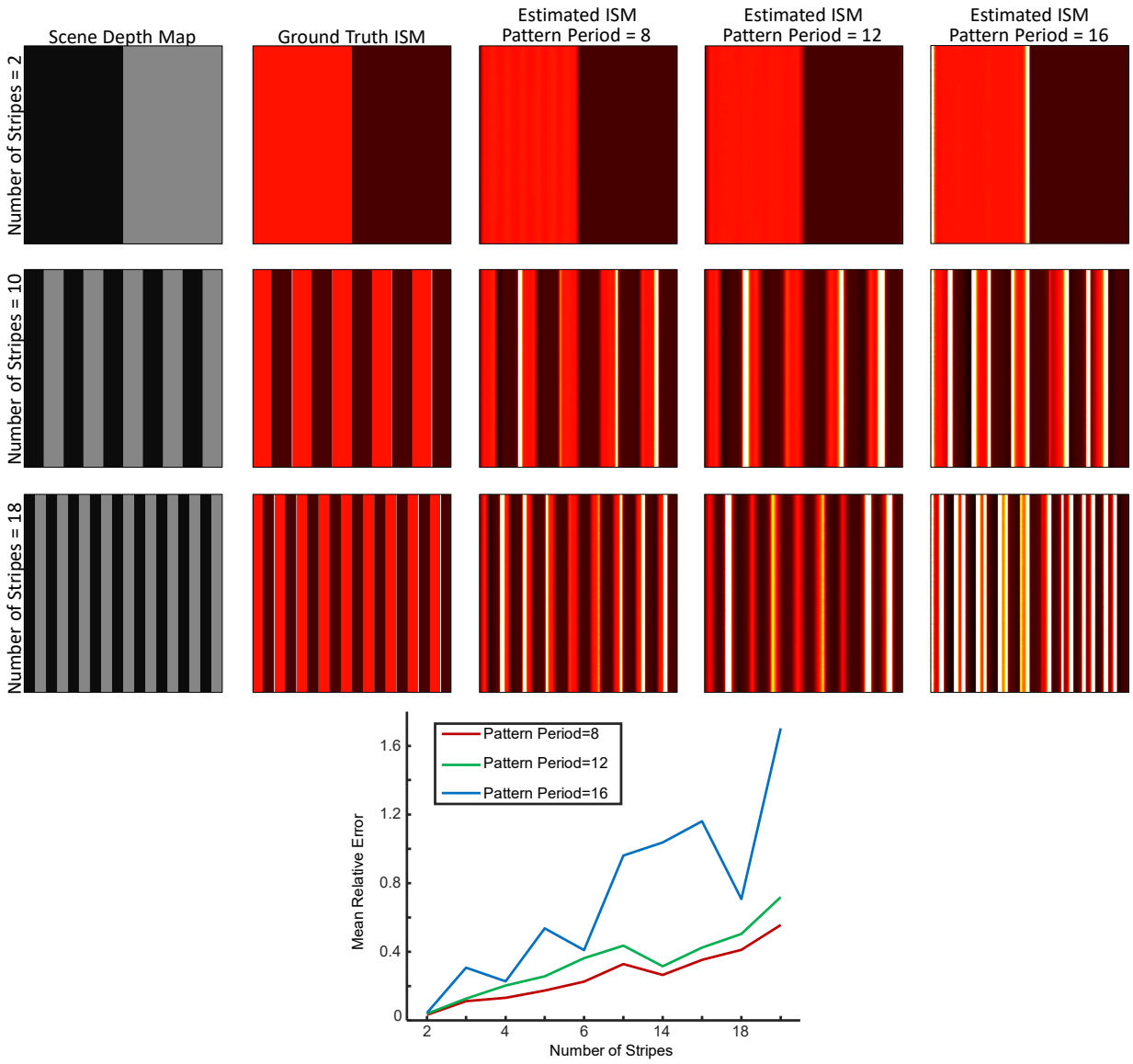

Figure 2: **Depth variation frequency vs. pattern frequency.** The scene consists of a series of vertical stripes at flipping depths. Camera moves in z-direction. The errors are higher for higher depth variation frequency. Higher pattern frequency can deal with the depth variation better and get higher accuracy.

1.2 Depth Variation Frequency vs. Pattern Frequency

Similarly, the ISM cannot be estimated when the depth varies too quickly. Fig. 2 shows the result for a scene consists of a series of vertical stripes of equal widths whose depths flip between a low value (1m) and a high value (2m). As the number of stripes increase, the depth varies more quickly, and the error in the estimated ISM becomes larger. Notice that for the pattern with lowest frequency, the error does not always increase because the estimated $\Delta\Upsilon$ at the depth edges are unpredictable, but the overall tendency is still increasing. Patterns with higher frequency can deal with the depth variation better and get more accurate results.

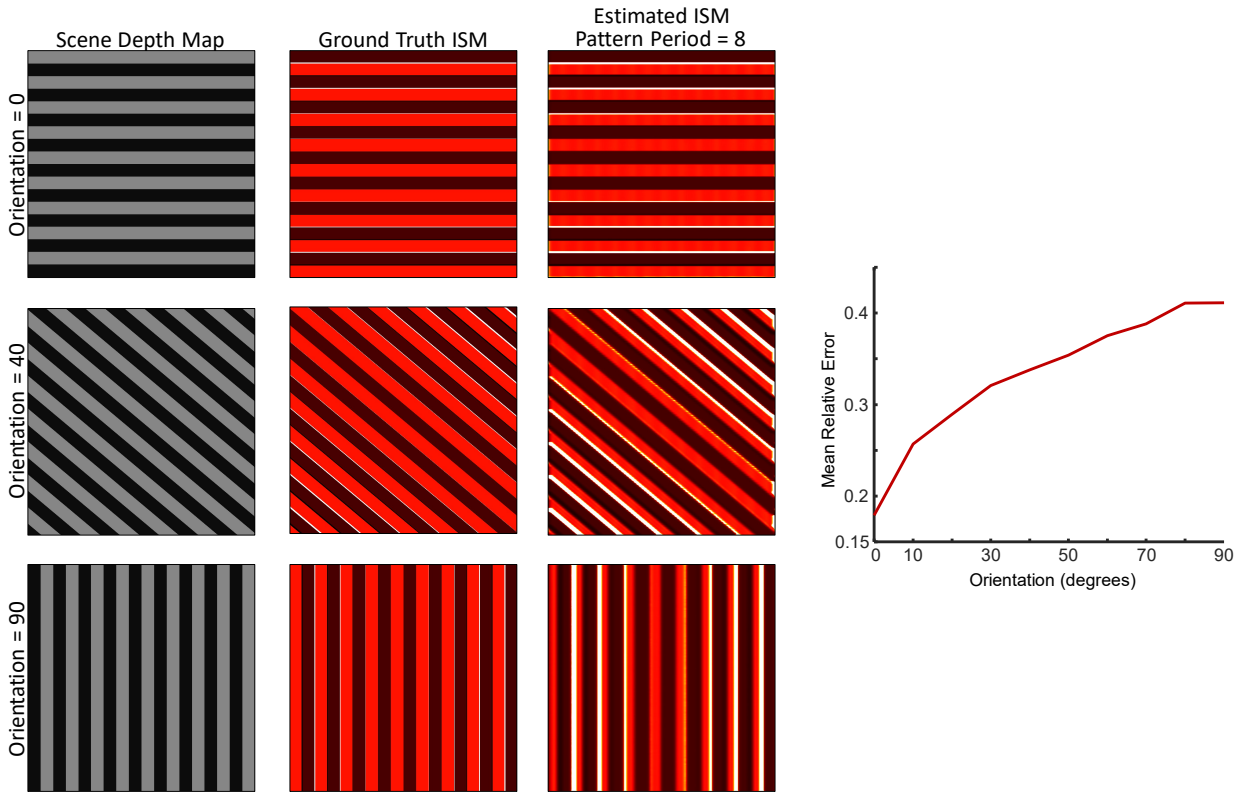

Figure 3: **Dependence on the orientation of depth edges.** We perform an in-plane rotation to the scene presented in Fig. 2. Orientation angles denote the angle between the edges and the horizontal direction. Errors are lower for nearly horizontal edges and higher for nearly vertical edges.

1.3 Orientation of Depth Edges

Section 1.2 studies the dependence of accuracy on the depth variation frequency. Notice that all the depth edges are vertical. An interesting question to ask is: Will the accuracy change if we keep the depth variation frequency, but rotate the depth edges to a different orientation? Fig. 3 plots the accuracy against the angles between the depth edges and the horizontal direction. 0° means the depth edges are horizontal, and 90° means the depth edges are vertical. The error increases as the orientation angle increases. This is because we assume a rectified camera-projector setup, and compute the Fourier transform along each row of the image. For horizontal edges, the depth does not change along the row, so the algorithm gets the best result. As the angle increases, the depth variation frequency along each row becomes higher, and the accuracy becomes lower. As a result, the accuracy is the lowest for vertical edges. Section 3.2 discusses a technique to improve the result for tilted depth edges (edges that are neither horizontal nor vertical).

1.4 Amount of Inter-frame Motion vs. Pattern Frequency

Fig. 4 studies the accuracy of the ISM estimate for different amount of inter-frame motion. The scene consists of a slanted plane, as shown in the depth map. The depth of the pixels ranges from 0.5m to 3m. Recall that the disparity change depends on the scene depth z and inter-frame z-motion Δz: $\Delta\Upsilon = -\frac{fb}{z^2}\Delta z$. When the motion is large and the scene depth is small, the disparity change can exceed half the period of the projection pattern and wrap around, and S_E will be incorrectly estimated as negative, as shown in the white region in

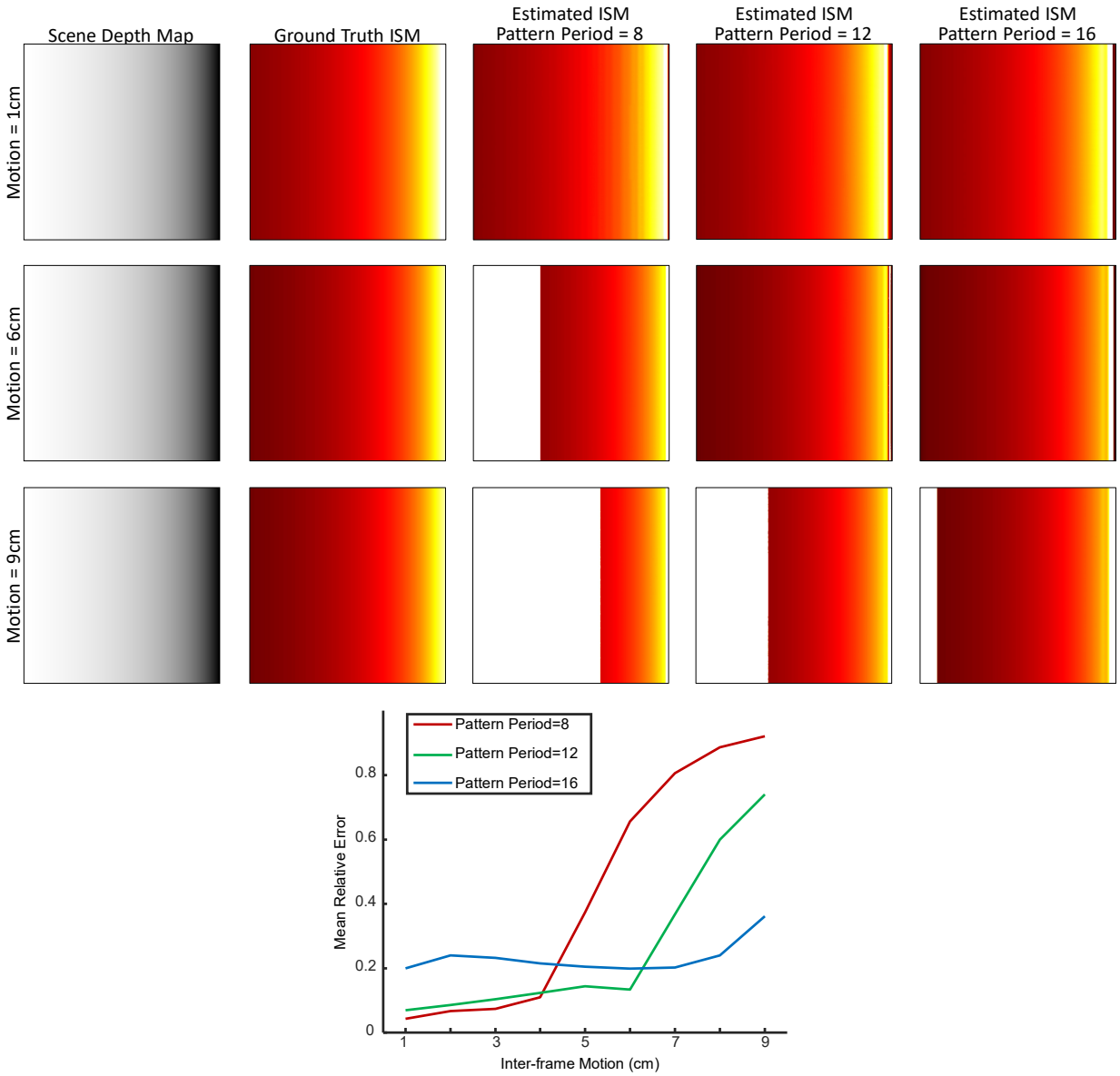

Figure 4: **Amount of inter-frame motion vs. pattern frequency.** The scene consists of a slanted plane. Camera moves in z-direction. Errors are larger for larger motion because there is a larger area where the disparity change exceeds the allowable range. For small motion, the high frequency pattern works better. For large motion, the high frequency pattern works worse because its allowable range of motion is smaller.

the estimated ISM. As the amount of motion increases, the area of the wrapped around region also increases, which results in a larger error. We can see that for small motion, higher pattern frequency performs better because it can resolves the smoothly varying depths better, but for large motion, there is a larger wrapped around region which means a higher error.

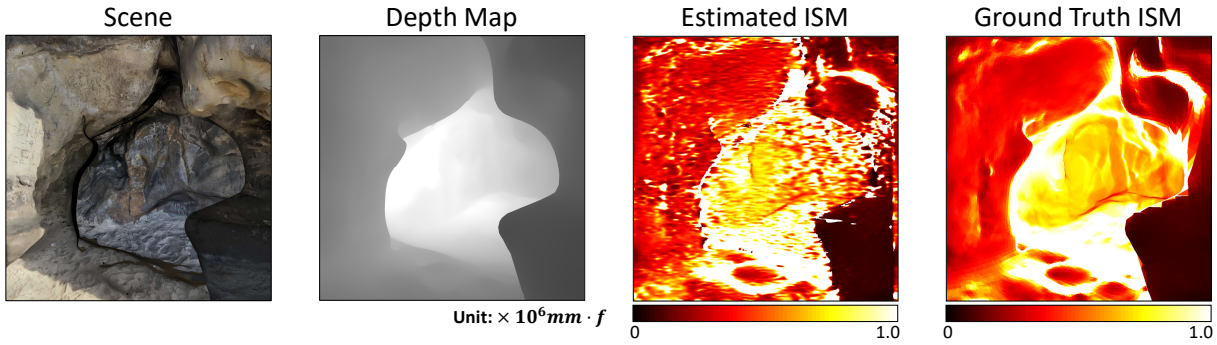

Figure 5: **Simulation results indicating real-world robot navigation scenarios.** The proposed method is able to recover the ISM despite the complex, high-frequency texture of a cave.

1.5 Dependence on Baseline

Same as all structured light and passive stereo systems, the maximum depth at which the proposed system works depends on the baseline. For a limited baseline, the proposed system is not able to estimate the ISM for objects at a very large distance, because the disparity change is too small to be estimated correctly. However, this is not a limitation of our method since only objects in the close range matter for collision avoidance applications.

2 Simulation and Experimental Results

In this section, we show the simulation and experimental results that are not included in the main paper due to space constraints.

Simulation results. Fig. 5 shows another simulation result that indicates a challenging scenario for real-world robot navigation. The proposed algorithm is able to estimate the ISM correctly despite the complex, high-frequency texture of the cave. The images (including those from the main paper) are generated by POV-Ray with an affine noise model.

ISMs for the same scene under different motion. Fig. 6 shows a tunnel built with planar cardboards. Our method recovers the significantly different ISMs due to different camera motion. In Fig. 6 (Center), the camera translates mostly along the z-axis. As a result, all three walls are getting close to the camera and thus, show a small ISM value. In Fig. 6 (Right), the camera translates mostly in $-x$ direction, which results in a very small ISM on the left wall and negative ISM on the right wall. The center wall and the ceiling also shows a finite, positive ISM because they are not perfectly aligned with the $xy-$plane and $xz-$plane.

3 Technical Details of the ISM Estimation Algorithm

3.1 Choice of Parameters

In this section, we discuss how to choose appropriate values for the parameters involved in our algorithm. One pair of parameters is the window size of the bandpass filter $(\omega_{uc}, \omega_{vc})$. Recall that we use a Hanning window for filtering in the frequency domain as suggested in [1],

$$H(\omega_u, \omega_v) = \begin{cases} \frac{1}{4}[1+\cos(\frac{1}{2}\pi\frac{\omega_u-\omega}{\omega_{uc}})][1+\cos(\frac{1}{2}\pi\frac{\omega_v}{\omega_{vc}})] & \text{if } (\omega_u, \omega_v) \in S_H \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

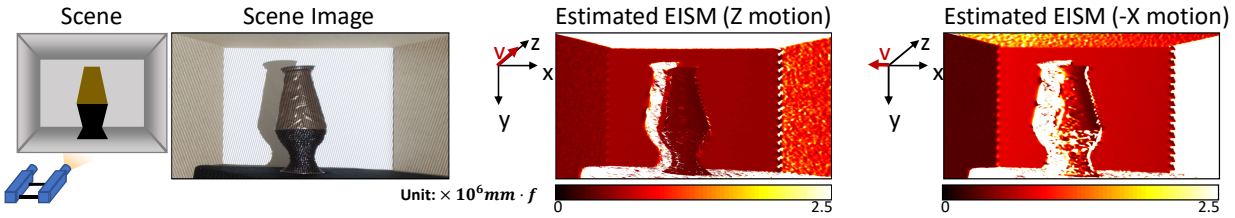

Figure 6: **ISMs for the same scene under different motion.** For the same scene (a cardboard tunnel with an "obstacle" in the center), our method is able to estimate the different ISMs for different camera motions. For z-motion, all three walls show finite ISM values. For $-x$-motion, the left wall shows a very small ISM, while the right wall has a negative ISM.

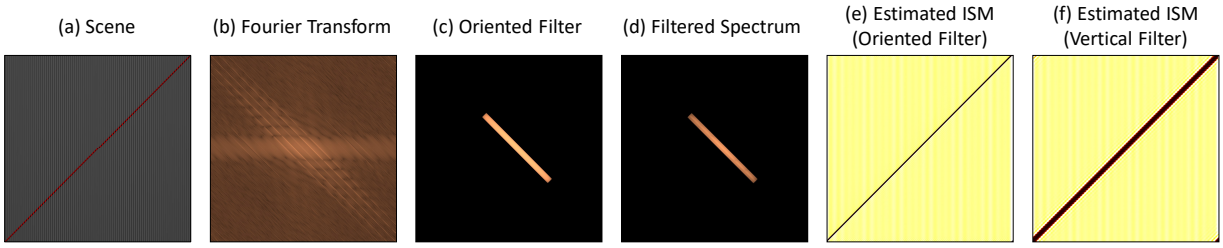

Figure 7: **Recovering thin structures by using an oriented filter.** (a) The scene depicts a thin thread in front of a wall. (b) Fourier transform of the image. Notice the tilted linear structures. (c) We can apply an oriented filter in order to separate the signal G. (d) Filtered spectrum. (e) Estimated ISM by the oriented filter. (f) Estimated ISM by the upright filter. The oriented filter generates more accurate results.

where ω is the angular frequency of the sinusoid, $(\omega_{uc}, \omega_{vc})$ is the cut-off frequency at a 50% attenuation ratio. S_H is the support for the filter:

$$S_H = \{(\omega_u, \omega_v) \mid |\omega_u - \omega| \leq \omega_{uc}, |\omega_v| \leq \omega_{vc}\}. \tag{4}$$

Using a larger support for the Hanning window will cover a larger range of the frequencies, which allows the recovery of more detailed structures at the cost of more sensitive to noise and the DC component of the spectrum. In our experiments, we choose $\omega_{uc} = \omega/2$ and $\omega_{vc} = \frac{\pi}{2N_v}$, where N_v is the height of the image.

Another parameter is the about post-processing the estimated ISM. After computing the ISM using the algorithm described in the paper, we use the estimated albedo map $|g|$ as a confidence map and filter out the values with low confidence. This step is used to exclude the pixels where no projection pattern is observed, which correspond to scene points in shadows or too far away from the projector. We set $S_E = +\infty$ when $|g| < 10^{-4}$, assuming the pixel intensities are normalized between 0 and 1.

3.2 Oriented Filter for Recovering Thin Structures

As shown in Section 1.3, tilted depth edges are harder to recover than horizontal depth edges. In this section we give details on how the oriented filter is designed to resolve tilted thin structures. Consider a tilted thin thread in front of a wall, which is shown in Fig. 7 (a). Since our method performs Fourier transform and filtering row by row, it is difficult to estimate the ISM correctly since the depth varies abruptly along each row. However, if we look at the Fourier Transform of the image (Fig. 7 (b)), tilted linear structures show up clearly in the spectrum, which correspond to the signals F, G and G^*. If we filter the spectrum using an oriented window, as shown in (c), instead of the vertical window we used in the main paper, it is still

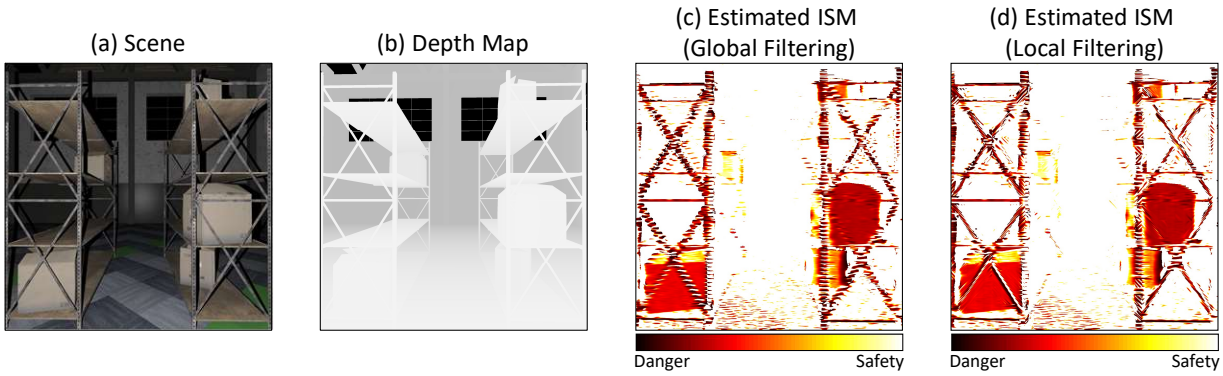

Figure 8: **Recovering complex geometry using local filtering.** Simulated warehouse scene with complex geometry. The local filtering method recovers the thin structures on the shelves more accurately.

possible to separate the signal G from F (d). As a result, the estimated ISM (e) is more accurate than using a vertical filter (f).

In practice, thin structures with different orientations can exist in the same scene, making ISM estimation difficult. One naive solution is to filter the spectrum using a discrete subset of the orientations (*e.g.*, $\{0, \pm\frac{\pi}{2}\}$), and then take a per-pixel minimum of the three obtained ISMs to give a conservative guess. A better way is to divide the image into small local patches and compute the Fourier transform of each patch individually. For each patch we choose the orientation of the filter that best separates the signal G from F, and only use that filter to compute the ISM of that patch. Neighboring patches need to be overlapped in order to get rid of artifacts near the patch borders. This local filtering helps resolving complex thin structures in the scene, as shown in Fig. 8.

The oriented 2D filter is defined as

$$H_{ori} = \begin{cases} \frac{1}{4}[1+\cos\frac{1}{2}\pi\frac{\omega_u\cos\alpha-\omega_v\sin\alpha-\omega\cos\alpha}{\omega_{uc}\cos\alpha}]\cdot[1+\cos\frac{1}{2}\pi\frac{\omega_u\sin\alpha+\omega_v\cos\alpha-\omega\sin\alpha}{\omega_{vc}\sin\alpha}] & \text{if } (\omega_u,\omega_v)\in S_{H_{ori}} \\ 0 & \text{otherwise} \end{cases}, \quad (5)$$

$$S_{H_ori} = \{(\omega_u,\omega_v) \mid |\omega_u\cos\alpha-\omega_v\sin\alpha| \le \omega\cos\alpha, |\omega_u\cos\alpha-\omega_v\sin\alpha| \le \omega\cos\alpha\}. \quad (6)$$

where α is the orientation of the filter. When $\alpha = 0$, this becomes the conventional vertical band-pass filter (Eq. 3). For a discrete subset of orientations $A = \{\alpha_1, \alpha_2, \dots \alpha_n\}$, we multiply the Fourier transform of the image $I(\omega_u, \omega_v)$ with another filter K_{ori} to detect the best orientation for H_{ori}, which is the orientation α_i with lowest energy response:

$$K_{ori} = \begin{cases} (\frac{\omega_u\cos\alpha-\omega_v\sin\alpha-\omega\cos\alpha}{b_u\cos\alpha})^2 & \text{if } (\omega_u,\omega_v)\in S_{H_{ori}} \\ 0 & \text{otherwise} \end{cases}, \quad (7)$$

This filter assigns higher weight to regions close to the boundary of the oriented filter H_{ori}. The intuition is that, if the spectra can be separated well, then the energy of $I(\omega_u, \omega_v)$ should be concentrated in the center of the support of the oriented filter, which results in lower total energy of $I(\omega_u, \omega_v) \cdot K_{ori}(\omega_u, \omega_v)$

Notice that the local filtering method does not increase the computational complexity of the algorithm. In fact, it slightly decreases the complexity. The computational complexity of the global filtering method is $O(M \log M)$, where M is the number of pixels. Suppose we have a patches, the complexity of filtering each single patch is $O(\frac{M}{a}\log\frac{M}{a})$. Then the complexity of the entire algorithm is $O(a \cdot \frac{M}{a}\log\frac{M}{a}) = O(M\log\frac{M}{a}$, which is slightly smaller than the global filtering method.

Although the oriented filter gives the method better ability of dealing with tilted thin structures, it does not help resolve vertical structures since there is no pattern variation along each column. As mentioned in the main paper, this limitation can be overcome by adding a second projector to form a L-shape configuration.

Patch size. The patch size for applying oriented filters is an important algorithm parameter. A small patch size causes loss of precision in the estimated spectrum. On the other hand, too large a patch may result in patches that contain multiple objects with different orientations, thus losing the benefit of oriented filters. Given this tradeoff, the optimal patch size depends on the frequency content of the image (texture, depth edges) and the expected object size in pixels (which depends on the resolution and field-of-view of the camera). In practice, we find patches with 128x128 lead to good performance. To remove unwanted boundary artifacts, the patches are overlapped and only the central 64x64 pixels are used.

4 Computational Efficiency

In the main paper, we compare the computing time for the proposed ISM estimation algorithm and a few widely-used stereo algorithms. Here we give more details on how the running time of those algorithms are profiled.

The MATLAB implementation of the proposed ISM estimation algorithm runs completely on CPU, which is compared with MATLAB's implementation of Semi-Global Matching [2]. The GPU implementation of the proposed ISM algorithm is implemented in C++ with CUDA Computing Toolkit 10.2. We compare the proposed algorithm with OpenCV's CUDA implementation of block matching, belief propagation [3] and constant-space belief propagation [4]. For block matching, the disparity range is chosen to be 1/20 of the image width. For the belief propagation methods, OpenCV's recommended parameter settings are used. For all GPU implementations, the computation kernel execution and the data exchange between CPU and GPU are carefully overlapped in time to maximize computing speed.

Since currently the DSLR is not able to capture continuous video frames and interface with the program, the captured sequences are processed offline. We assume all video frames are already loaded into the system RAM before execution of the algorithms. In a practical system the frames can be streamed from the camera to the RAM, which can be done in parallel with the computing process and should not take additional time.

We test the methods on a laptop with a Intel Core i7-7700HQ CPU and a NVIDIA GTX 1060 MAX-Q GPU. All methods are tested on simulated images with different resolutions. At each resolution 100 images are processed and the average running time per frame is reported. Our GPU implementation is able to process 1000 frames per second at 1 megapixels. This is comparable with state-of-the-art single-shot structured light depth recovery algorithms [5], though their algorithm is profiled on a better GPU. We are unable to compare with their method directly because their code is not publicly available. At very high resolution (9 megapixels), the proposed method is still able to run in real-time (90fps), which is 9x faster than block matching.

The proposed method has shown consistent computational benefits across computing architectures. Notice that in practice, powerful GPUs are usually not available on a micro drone. The actual implementation of ISM algorithms should be adapted to the available computing resources for different applications.

5 Experiment Setup and Calibration

Fig. 9 (Left) shows the setup of our hardware prototype. We rigidly fix our Canon DSLR camera and Epson 3LCD projector together. The whole system is placed on a cart such that it can move freely in x and z directions (*i.e.*, in the horizontal plane). We calibrate our system by placing a checkerboard at different locations and orientations, capturing images and computing the intrinsic and extrinsic matrices using standard stereo calibration algorithm. The correspondences between scene points and projector pixels are established by projecting a series of phase-shifting sinusoids.

The derivations and algorithm in the main paper are based on the assumption that the camera-projector system is rectified. Fig. 9 (Right) demonstrates image rectification for passive stereo. A scene point R is observed by two cameras O_L and O_R. The original images i_L and i_R are rotated by the homographies H_L and H_R into the rectified images i'_L and i'_R such that the epipolar lines (red dashed lines) are horizontal. In a structured light system, the left camera O_L is replaced by a projector. i_L is the pattern actually projected

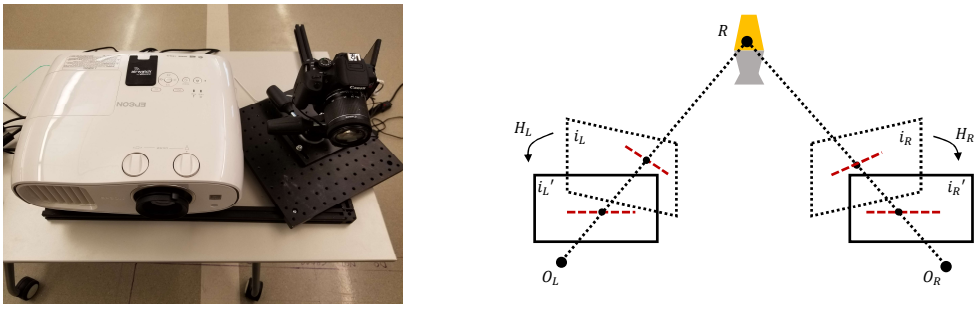

Figure 9: **(Left) Hardware setup.** The Canon DSLR camera and Epson 3LCD projector are rigidly attached together. The system is placed on a cart that can move in x and z directions. **(Right) Image rectification for stereo systems.** A scene point R is observed by two cameras O_L and O_R. i_L and i_R are the raw images. i'_L and i'_R are the rectified images, which are obtained by applying homographies H_L and H_R to the raw images. Epipolar lines for the images are shown as red dashed lines. In our structured light system, the projected pattern need to be transformed by the inverse homography H_L^{-1} before projection.

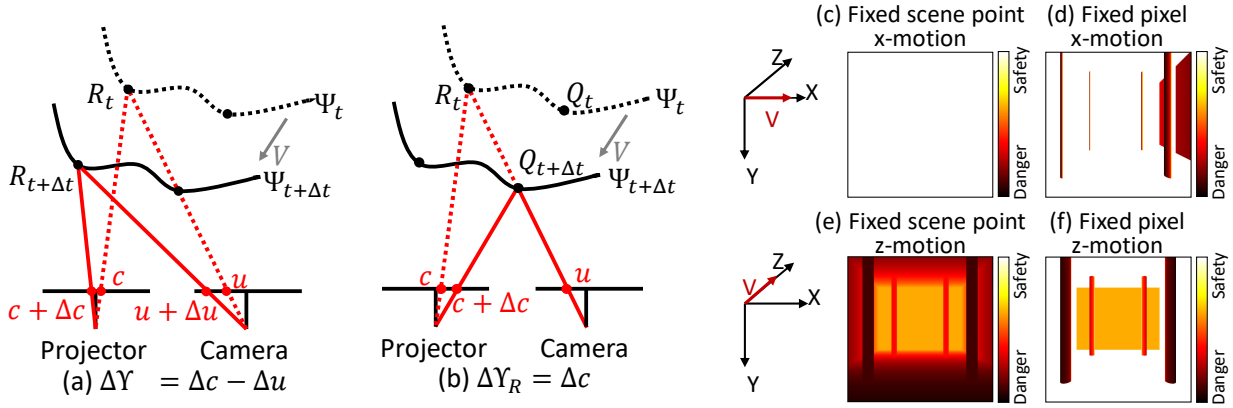

Figure 10: **Fixed-pixel disparity change. (a) Fixed-scene-point definition.** Scene surface Ψ moves relatively to the camera between time t to $t + \Delta t$. The same scene point R is tracked from t to $t + \Delta t$. $\Delta\Upsilon_L$ is computed as the difference of disparities of R_t and $R_{t+\Delta t}$. **(b) Fixed-pixel definition.** $\Delta\Upsilon_E$ is computed from two different points R_t and $Q_{t+\Delta t}$, both projecting to the same pixel u. **(c-f) Comparison of the two definitions for the same scene.** In (d), ISM$< +\infty$ on the walls and pillars because these surfaces are not fronto-parallel and the fixed-pixel depths decrease as the camera moves to the left. In (f), ISM$= \infty$ on the walls because the camera moves in parallel to the walls and the fixed-pixel depths do not change. Such differences need to be taken into account for navigation policy design.

by the projector and i'_L is the rectified pattern. Therefore, different from the passive stereo case, we need to perform an "unrectification" in order to get the actual projection pattern. We first choose the rectified pattern i'_L to be a 1D sinusoid pattern, with all the pixels in the same column having the same intensity. The projected pattern i_L is then computed by apply the inverse homography H_L^{-1} to i'_L.

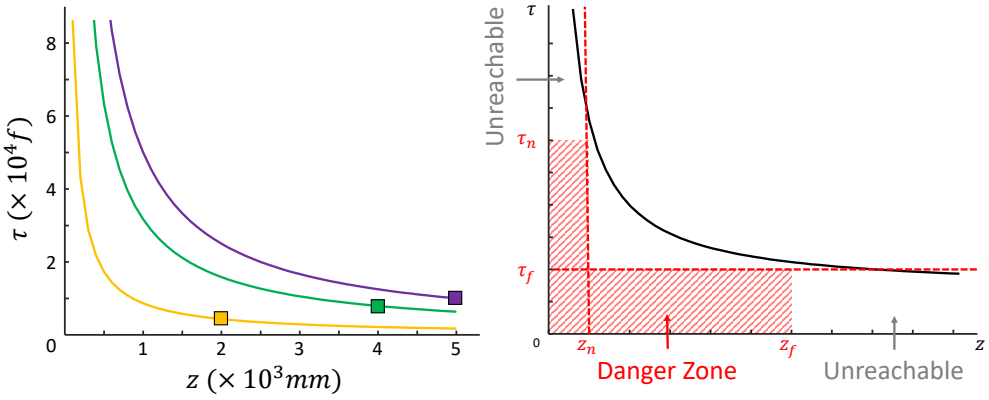

Figure 11: $z-\tau$ **plot. (Left)** A given value of the ISM corresponds to an infinite number of possible $z-\tau$ pairs lying on a hyperbolic curve called the $z-\tau$ curve in the 2D $z-\tau$ space. **(Right)** We can specify danger zones in the $z-\tau$ space based on application specifications; potential danger can be identified by checking whether the $z-\tau$ curve intersects with the danger zone.

6 Fixed-Pixel Disparity Change

Fig. 10(a-b) shows the difference between the fixed-scene-point disparity change and fixed-pixel disparity change definition. In the fixed-scene-point definition, we track the same scene point R before and after motion. The disparity change is given by $\Delta\Upsilon = \Delta c - \Delta u$, which depends on both the camera image motion Δu and the change in projector column index Δc. In the fixed-pixel definition, we consider the disparity change along the same camera ray that corresponds to pixel u, which projects to different scene points before and after motion (R_t and $Q_{t+\Delta t}$). The disparity change is the same as the change in projector column index: $\Delta\Upsilon_R = \Delta c$, which is intuitively easier to compute.

Fig. 10(c-f) shows the computed ISM for both fixed-scene-point definition and fixed-pixel definition. As mentioned in the main paper, fixed-pixel definition may result in overly conservative collision warnings. This is because the fixed-pixel definition tracks the disparity change at a fixed camera pixel. For example, in Fig. 10(c,d), although there is no z-motion, the ISM deems the edges of the pillars as dangerous pixels due to pixelwise depth changes. These pixels are false positives because, if the camera keeps moving horizontally, it will not crash into the pillars. We can mitigate this issue by spatio-temporally filtering the ISM, since these pixels are usually along depth edges and their depths will not continue to decrease indefinitely as the camera moves. If the depth of a pixel does keep decreasing, then there is a true danger of collision (right vertical walls in the top row). In practice we apply a 5×5 median filter spatially and a 5-frame box filter temporally.

In general, such false positives exist for fast x/y-translation and rotation. In such scenarios, ISMs should not be used as the only collision avoidance measure but should work as a warning mechanism that triggers a more accurate, yet time-consuming depth recovery process.

7 Potential Policies Based on the $z-\tau$ Curve

As mentioned in Section 3.2 in the main paper, for a given value of ISM, the possible pairs of (z, τ) lie on a hyperbolic curve called $z-\tau$ curve. Although it is generally not possible to recover z and τ directly, we can still design conservative navigation policies to enable fast collision avoidance. One naive policy is to set a warning threshold for the ISM. A more principled option is to specify *danger zones* in the $z-\tau$ space based on the applications and robot specifications (Fig. 11), and then check if the $z-\tau$ curves intersect with the danger zone. For every pixel in the image, if its corresponding $z-\tau$ curve intersects with the danger zone,

then the robot may crash into the corresponding scene surface and therefore may want to speed down or plan its trajectory to circumvent the obstacles accordingly.

One way to determine the danger zone is to set a threshold of time to contact τ_f (red dashed line in Figure 11(Right)) which can be interpreted as the minimum amount of time for the robot to react and avoid collision. τ_f can be potentially determined based on the mobility specifications of the robot such as the maximum speed and acceleration. Notice that only part of the region below the red line is marked as danger zone because a combination of a small τ and a large z is impossible due to the physical limit of the robot speed. For a fixed τ_f, the maximum reachable z_f is determined by

$$z_f = \tau_f \cdot v_{max} \,, \tag{8}$$

where v_{max} is the maximum speed of the robot.

Another part of the danger zone comes from a threshold of depth d_n. Being too close to a surface can be dangerous even if the time-to-contact is large due to panning motion, because there is still a chance of collision due to unpredictable external forces. For example, a micro drone may be easily affected by strong winds. Notice that a combination of small z and extremely large τ is also marked as unreachable. This is because in practice completely panning motion is almost impossible to maintain. If it does occur then we can argue that the panning motion can be continued stably, so there is no danger.

By adding these extra constraints, we get a finite danger zone. We can then determine whether a pixel is *dangerous* by testing if the $z - \tau$ curves intersect with the danger zone. This provides a conservative policy since there are no false negatives for dangerous pixels. To deal with false positives, an adaptive strategy can be deployed. When dangerous pixels are detected, we can lower the speed of the robot to allow more reaction time, during which we run a more time-consuming depth estimation algorithm to determine whether it is actually a danger or a false positive. We can also estimate the probability of being a dangerous pixel by computing the fraction of the curve that is inside the danger zone to balance the false positive rate and false negative rate.

We give two example ways of specifying danger zones in this section. In practice, a danger zone can be any arbitrary subset of the entire $z - \tau$ space, depending on the exact constraints for the application scenario. We expect more sophisticated navigation policies to be designed and deployed in real applications in future work.

References

[1] J.-F. Lin and X.-Y. Su, "Two-dimensional Fourier transform profilometry for the automatic measurement of three-dimensional object shapes," *Optical Engineering*, vol. 34, p. 3297, Nov. 1995.

[2] H. Hirschmuller, "Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, (San Diego, CA, USA), pp. 807–814, IEEE, 2005.

[3] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Belief Propagation for Early Vision," *International Journal of Computer Vision (IJCV)*, vol. 70, pp. 41–54, Oct. 2006.

[4] Q. Yang, L. Wang, and N. Ahuja, "A constant-space belief propagation algorithm for stereo matching," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, (San Francisco, CA, USA), pp. 1458–1465, IEEE, June 2010.

[5] S. R. Fanello, C. Rhemann, V. Tankovich, A. Kowdle, S. O. Escolano, D. Kim, and S. Izadi, "HyperDepth: Learning Depth from Structured Light without Matching," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (Las Vegas, NV, USA), pp. 5441–5450, IEEE, June 2016.