Simulating Kinect Infrared and Depth Images

Michael J. Landau, *Member, IEEE*, Benjamin Y. Choo, *Member, IEEE*, and Peter A. Beling, *Member, IEEE*

***Abstract*—With the emergence of the Microsoft Kinect sensor, many developer communities and research groups have found countless uses and have already published a wide variety of papers that utilize the raw depth images for their specific goals. New methods and applications that use the device generally require an appropriately large ensemble of data sets with accompanying ground truth for testing purposes, as well as accurate models that account for the various systematic and stochastic contributors to Kinect errors. Current error models, however, overlook the intermediate infrared (IR) images that directly contribute to noisy depth estimates. We, therefore, propose a high fidelity Kinect IR and depth image predictor and simulator that models the physics of the transmitter/receiver system, unique IR dot pattern, disparity/depth processing technology, and random intensity speckle and IR noise in the detectors. The model accounts for important characteristics of Kinect's stereo triangulation system, including depth shadowing, IR dot splitting, spreading, and occlusions, correlation-based disparity estimation between windows of measured and reference IR images, and subpixel refinement. Results show that the simulator accurately produces axial depth error from imaged flat surfaces with various tilt angles, as well as the bias and standard lateral error of an object's horizontal and vertical edge.**

***Index Terms*—Computer-aided design (CAD), infrared (IR) dot pattern, Microsoft Kinect, simulation, speckle noise, structured-light 3-D scanner.**

I. Introduction

THE MICROSOFT Kinect has become the most popular and widely used consumer-grade depth sensor for research purposes, which is apparent in the vast number of publications with Kinect data, as well as the total sales surpassing 24 million units. There are also several well-established developer communities sharing resources that involve the use of Kinect, such as the robot operating system and the Microsoft developer network. And since the sensor has a simple and inexpensive optical design, the cost of Kinect has been driven down so much that nearly any research group across many disciplines can have access to a reliable and fairly accurate depth sensor for a variety of applications. These applications include, but are not limited to, using the sequence of human poses for activity recognition [1], scene classification [2], autonomously navigating [3] or reconstructing [4] environments with an unknown layout, and 3-D object [5] and pose estimation [6].

Any application not involving the estimation of human pose will generally involve inferencing attributes of inanimate and rigid objects that can potentially be represented by computer-aided designs (CADs). Since Microsoft only provides tracking software for human pose in its software development kit (SDK), researchers must rely on utilizing the raw depth images of the objects produced from the Kinect device [7]. Thus, in order to test the effectiveness of a new system that processes output depth data streams, an appropriately large ensemble of experiments with accompanying ground truth is required. This is, however, not always feasible, or at least proves to be an arduous task that may require the assistance of imprecise or expensive tools to gather the imperfect "truthing" data. There are also extensive data sets of objects recorded by the Kinect sensor that are made available online to avoid the tedium of estimating a ground truth, such as the depth data sets provided by Berkley, New York University (NYU), Princeton, and the University of Washington (UW).[1] These data sets only cover a few specific objects and scenes, however, and as such, users are limited to testing their model on depth images not necessarily related to their specific goals. Furthermore, the online resources listed as well as many other available data sets do not include a model of the recorded object, and the user is left to find a CAD model of a related object close in shape and size, or use a 3-D object scanning method [8] to generate a less than perfect model.

These limitations motivate the need for a practical, reliable, and extensible Kinect infrared (IR) and depth image simulator with included built-in speckle and noise models. Several general depth sensor simulators already exist, such as the Georgia Tech LADAR simulation developed by Dixon,[2] which allows the user to modify the sensor specifications and construct a scene with CAD models. However, the Kinect sensor implements specialized Light Coding technology in accordance with a unique transmit IR dot pattern, and therefore estimates noisy depth images atypical to that of any other structured-light or range imaging sensor. The Microsoft robotics developer studio (RDS) provides a simulated Kinect sensor that estimates depth and RGB images within the limited depth range of the real Kinect. The RDS software focuses

Manuscript received April 7, 2015; revised August 17, 2015; accepted October 11, 2015. Date of publication November 13, 2015; date of current version November 15, 2016. This paper was recommended by Associate Editor L. Shao.

The authors are with the Department of Systems and Information Engineering, University of Virginia, Charlottesville, VA 22903 USA (e-mail: mjl5b@virginia.edu; byc6j@virginia.edu; pb3a@virginia.edu).

This paper has supplementary downloadable material provided by the author and made available on MATLAB Central's File Exchange Resource Center, which is titled ⟨Kinect Infrared (IR) and Depth Image Simulator⟩. The material includes code for the Kinect simulator as well as a demo.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2015.2494877

[1] See: Berkley ⟨kinectdata.com⟩; NYU ⟨cs.nyu.edu/~silberman⟩; Princeton ⟨vision.princeton.edu⟩; and UW ⟨rgbd-dataset.cs.washington.edu⟩

[2] J. H. Dixon, *LADAR Simulator User Manual*, 2007. ⟨jdixon@ece.gatech.edu⟩

mainly on simulating the sensor on a mobile platform, though, and lacks the granularity of Kinect's inherent depth image noise and error characteristics. More specifically, synthetic sequences are generated by simply rendering ideal images of 3-D objects and adding homogeneous Gaussian noise. Therefore, this paper's focus is on a high fidelity Kinect simulator that constructs noisy IR and depth images of user supplied CAD models by modeling the physics of the transmit and receive optics, the unique dot pattern, and by closely following the actual methods and algorithms that the real Kinect sensor employs.

Our IR and depth image simulators not only involve newly constructed models, but also take advantage of methods and tools previously developed from other projects, and couple them with well-defined Kinect sensor mechanics and specifications. For instance, we make use of an idealized bitmap representation of the Kinect dot pattern [9] as well as an optimized ray casting method [10] to simulate line-of-sight vectors originating from Kinect's IR laser transmitter. By modifying these tools, we are able to simulate IR dot splitting and spreading, as well as random intensity variation and IR noise. These idiosyncrasies distort the measured pattern and contribute to Kinect's unique error composition, and are, therefore, a necessary inclusion for a fidelitous simulator. Also, since the IR image is received by a sensor offset by a known baseline distance from the transmitter to perform stereo triangulation, we can in effect emulate dot occlusions in the IR image and depth image shadowing that results from the surfaces of objects separated at a distance from its background. Moreover, since the embedded Light Coding technology implements a correlation-based method to match local sections of the sensed dot pattern to templates from a reference image, a match is not well-defined when part of the IR pattern is occluded or when segments of the pattern are horizontally offset. This leads to erratic and inhomogeneous depth estimation errors that cannot be faithfully modeled by constructed point cloud covariances that bypass IR imaging and subsequent disparity/depth processing. More specifically, depth estimation error of a measured Kinect image is largely dependent on the orientation of the object and surrounding scene, where error tends to grow larger for pixels that sense near an object edge or tilted surface.

The rest of this paper is structured as follows. Section II-A provides a thorough review of the Kinect hardware and software specifications, and Section II-B gives an overview of several attempts to generate empirical models that estimate axial and lateral error. We then construct a set of empirical models for random intensity speckle of the dots and IR noise in the detectors in Section III. Together, the ensemble of information is used to construct the proposed Kinect simulator in Section IV, which models disparity estimation to construct depth images from simulated IR images. It should be noted that the simulator focuses on first principles, and therefore does not model post-processing filters to improve depth estimates. Example results of the simulator are then presented in Section V, which are compared to experimental data and existing error models to validate the accuracy. Finally, we suggest possible applications and directions for future work in Section VI, and summarize our findings in Section VII.

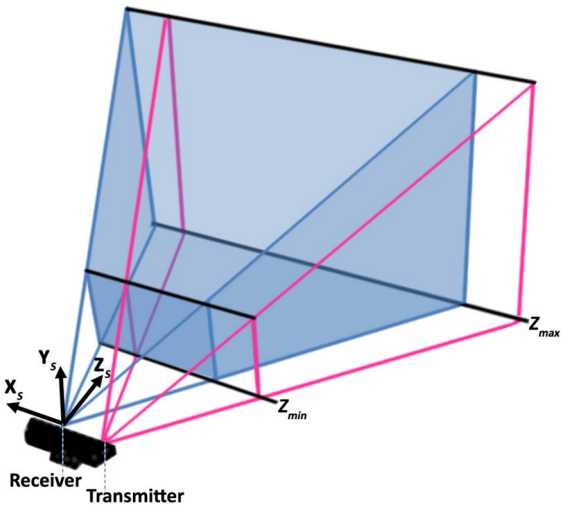


Fig. 1. The Kinect transmitter and receiver position and orientation are displayed, as well as the minimum $Z_{\min}$ and maximum $Z_{\max}$ operational depths. The sensor coordinate system is collocated with the receiver coordinate system, and the X-axis $\mathbf{X}_S$ and Z-axis $\mathbf{Z}_S$ of the sensor coordinate system are, respectively, collinear and parallel with the corresponding transmitter coordinate system axes.

II. Background Literature

A. Background of the Microsoft Kinect Sensor Mechanics

In this section, we provide a detailed review of the Microsoft Kinect sensor's underlying mechanisms and performance characteristics based on existing literature, which in turn motivates the design for our constructed simulation models. When the Kinect was first released on November 2010, it was initially intended as a human gesture recognition controller in the gaming context. Since then, because it is small-sized and commodity-priced, its usefulness as a 3-D depth sensor was rapidly recognized within many third party applications, and on February 2012, a version for Windows was released. Before the initial release, PrimeSense—the company that conceived the hardware design and camera sensor chip used in the Kinect—first published a patent [11] for the design of a system allowing for the capability of real-time object reconstruction by depth mapping with projected dot patterns. Many additional patents [12]–[16] were then issued and made available to the public domain that detail several structured light and projection system designs to triangulate and estimate a scene's depth data. However, much of the sensor implementation details were still left undisclosed. In the past several years, great lengths have been taken to understand the device[3] in order to surmise exactly how depth images are generated using PrimeSense's patented Light Coding technology.

The Kinect system makes use of two devices to construct depth images: 1) a class 1M IR laser and 2) an Aptina MT9M001 complementary metal-oxide-semiconductor (CMOS) sensor [17], [18], which are coplanar and aligned to have parallel optical axes, and are offset by

[3]See: blog sites ⟨hackaday.com⟩ and ⟨futurepicture.org⟩ for multiple examples of Kinect "hacking"

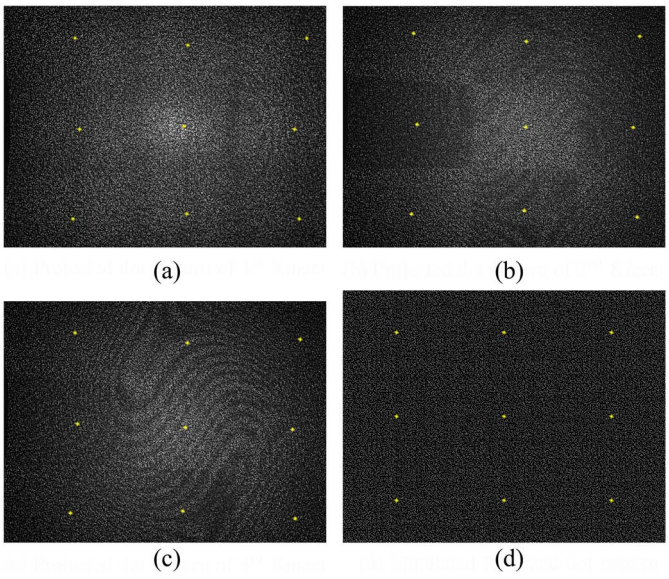


Fig. 2. Dot patterns of a Kinect for Windows (a) and two Kinects for Xbox (b) and (c) are projected on a flat wall from a distance of 1000 mm. Note that the projection of each pattern is similar, and related by a 3-D rotation depending on the orientation of the Kinect diffuser installation. The installation variability can clearly be observed from differences in the bright dot locations (yellow stars), which differ by an average distance of 10 pixels. Also displayed in (d) is the idealized binary replication of the Kinect dot pattern [9], which was used in this project to simulate IR images.

a baseline distance of 75 mm [19]. The laser projector and monochrome IR image sensor both work in stereo as a matricial active triangulation system where a pseudorandom IR dot pattern is transmitted and received, respectively. According to [20] and stated as an assumption in [12], the origin of the 3-D sensor coordinate system is centered exactly over the position of the receiver's image capture assembly with coordinates $(0, 0, 0)$ mm, which places the center of the optical axes of transmit illumination at coordinates $(-75, 0, 0)$ mm. In this regard, the terms receiver coordinate system and sensor coordinate system are used interchangeably. The Kinect for Windows v1 SDK [21] reports the angular field of view (FOV) of the sensor as 58.5° and 45.6° along the horizontal and vertical axes, respectively. The SDK also reports the default operational depth range between 800 and 4000 mm, though the hardware can be switched to near mode, which provides a range of 500 and 3000 mm. For a visual depiction of the transmitter and receiver position, orientation, and FOV, refer to Fig. 1.

The laser projection system, referred to as the transmitter, contains the illumination unit that produces a coherent light source, and a diffuser unit that generates and propagates a pseudorandom, uncorrelated dot pattern onto a region in space. The illumination unit consists of an IR light-emitting diode which generates a beam in a relatively narrow band of wavelengths [14]. The diffuser unit contains two diffractive optical elements (DOEs) arranged in series, which are comprised transparencies that act as active optical surfaces to diffract the input IR beam [15], [16]. A positive image of the dot pattern on the first DOE generates the uncorrelated distribution of bright and dark spots so that the auto-correlation of local patterns on the transverse plane is minimal. The second DOE then tiles the diffracted beam into a 3×3 grid of output beams with a specified fan-out angle to at least partially cover the region of space within the FOV of the imaging unit. According to [15], since the receiver's FOV is within a certain constraint, the second DOE is constructed to provide nearly perfect tiling with minimal overlap and/or gaps of the transmitted dot pattern. The diffraction pattern in the center tile suffers slight barrel distortion, however, and the outer tiles suffer significant pincushion distortion [16]. It should be noted that while the dot pattern is common to all Kinect sensors, small discrepancies in the factory installed diffuser unit result in differences between the projected patterns of each sensor when placed at the same position and orientation [Figs. 2(a)–(c)]. The yellow stars in Fig. 2 represent bright dots on the projected surface, which result from an excess of IR light that is passed through and propagated by the system—this is a side effect of the zero-order beam problem [13].

The laser imaging system, referred to as the receiver, contains the CMOS active-pixel digital sensor with 1280 (columns) $\times$ 1024 (rows) of active pixels and a 10-bit analog-to-digital converter (ADC) on-chip, which records images at 30 frames/s [22]. A bandpass filter positioned near the lens on the receiver assembly allows light only in the IR spectrum to be detected, while filtering out most of the ambient light that would otherwise reduce the contrast of the image [15]. Accordingly, each photoconductor in the sensing array receives different sections of the transmitted IR dot pattern, where photons are interpreted as a current or voltage drop [23], and quantized into integer intensity values between 1 and $2^{10} = 1024$ for each pixel on the imaging platform. Though the size of the dots increases with an increase in depth, the ratio remains constant, and for ideal surfaces the average dot size is designed to remain at least two pixels wide [11]. It is worth noting here that prior to depth mapping, Kinect uses 2×2 binning to downsample the IR image to 640×512 pixels, where a transmit dot then fills a single pixel on an ideal surface. Also, since the propagated beam has a relatively large depth of focus, high contrast IR images are maintained over the operational range of depths [13]. A predetermined threshold presumably filters and converts the IR intensity images to binary images to allow for faster processing.

Since the pseudorandom dot pattern is not time varying and does not vary along the Z-axis, Reichinger was able to construct a single image of an idealized binary replication by capturing the IR pattern with a camera [9]. The term idealized is used to imply a theoretically perfect factory installation of the diffuser unit that generates and propagates the dot pattern, and as such does not portray a 3-D rotation when projected on a flat wall. Moreover, it is assumed that the tiled pattern forms a perfect 3×3 grid of an initially generated 211×165 subpattern of bright and dark spots, where nonlinear distortions are removed. Also, for simplicity, the shape and size of each dot fills a single cell (pixel) of the constructed grid. The final image [Fig. 2(d)] has a resolution of 633×495, where $3861/34,815 = 11.09\%$ of the pixels contain a dot. This is consistent with the low duty cycle requirement enforced

TABLE I
UNIQUENESS TEST FROM CORRELATION WINDOWS WITH VARYING ROWS (R) AND COLUMNS (C)

R \ C	5	7	9	11
5	43130	6152	838	91
7	5900	352	13	0
9	810	18	0	0
11	133	0	0	0

in [15], where the fraction of bright spots should not exceed the (1/10)th area of the total projected pattern in order to have good performance in the depth mapping process.

All of the defining properties of Kinect's spatially fixed dot pattern allow for reduced optical and computational complexity for object reconstruction [11]. Therefore, with the advent of PrimeSense's Light Coding object reconstruction technology, all 3-D information of a scene is processed in real time on the company's PS1080 system-on-chip (SoC), another device integrated within the Kinect casing [17]. Object reconstruction is performed by first estimating the horizontal shift or disparity of a local window of the dot pattern on the received image, and then computing the depth z from disparity d using the standard stereo-triangulation formula [23]

$$z = f_x \frac{b}{d} \tag{1}$$

for each of the 640×480 downsampled pixels. Here, b represents the baseline distance between the transmitter and receiver, and f_x represents the horizontal focal length of the receiver, measured in pixels. The size of the correlation window has been determined to be 9×9 pixels based on observation and analysis of various IR image samples [19]. This theory can be further supported by constructing correlation windows of various sizes, centering them on each pixel within the subpattern of the idealized dot pattern, and comparing them to the remaining pixels within the same row (Table I). A 9×9 window size is the smallest in order to achieve no pattern overlap and preserve a unique code.

Disparity estimation is accomplished in two steps. The first step determines the best match between a spatial multiplexing window centered on a pixel of the measured IR image and a window of a reference IR image. Since the transmitter and receiver are rectified to have corresponding horizontal epipolar lines, the search can be limited to the same row of pixels. A series of reference images, which represents the pattern projected on planar surfaces parallel to the optical axis at various depths, is precaptured and stored on the memory utility of the system. In order to find the best match between the measured and reference images, the cross-correlation values of each window pair is maximized [12]. This step is inherently robust to a degree of lens and perspective distortion, though the disparity estimate is limited to an integer pixel shift corresponding to the correlation peak. Furthermore, cross-correlation works best on flat surfaces that are parallel to the focal plane, but suffers error when the dot pattern is significantly distorted for windows that span significant depth variation. Though it is not explicitly stated, [11] suggests that a prediction-based region growing algorithm that interpolates depth values from neighboring pixels provides a good tradeoff between complexity and performance. Thus, this method is also most likely performed to disambiguate potential matches and improve the quality of depth estimates on pixels with correlations below a threshold.

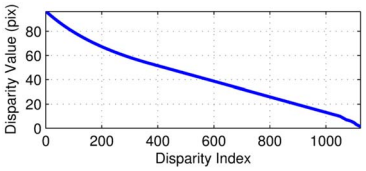


Fig. 3. The disparity values were computed by using (1) with all quantized Kinect for Windows depth values.

The second step in disparity estimation performs a subpixel refinement to determine how much the IR dots of the window are splitting between adjacent receive pixels, which further improves the estimated disparity accuracy. Based on the observation of Kinect output disparity values done by [19] and [24], the Kinect decidedly estimates disparity to a resolution of (1/8)th of a pixel. Concordantly, by plotting the estimated disparity from the array of all quantized Kinect depth values, the data show a linear trend with a slope of roughly $(-1/8)$ pixels (Fig. 3). This is consistent with the subpixel refinement interpolation factor, which means Kinect provides quantized 11-bit depth values contingent on all estimable subpixel disparity values.

B. Related Work for Kinect Error Models

Several reports have been published that model the 3-D spatial errors exhibited by point clouds extracted from Kinect depth images. In these reports, various geometric, empirical, and statistical models are developed for axial and lateral error. In this context, the axial component refers to the distance along the depth or Z-axis and the lateral component refers to distances orthogonal to depth along the X- and Y-axis of the focal plane. A thorough compilation of independent analyses that characterize error and noise types in Kinect point clouds can be found in [25]. Here, Mallick *et al.* [25] provided a uniform nomenclature for different error types and models, which are used for reference in this paper.

An early investigation of the Kinect for Xbox composed by Menna *et al.* [24] reports the sensor's lateral and depth error models based on basic geometric constraints of a typical triangulation system. More specifically, the model assumes a pinhole camera system specified only by the intrinsic parameters of pixel size and focal length of the receiver, and the baseline distance. Their proposed standard error models account for predicted quadratic and linear errors that exist in the axial and lateral components over the range space of Kinect. However, such geometric models have since been deemed too simplistic, and more sophisticated models were proposed that account for other key influential factors.

In contrast, Nguyen *et al.* [26] developed empirical models for axial and lateral error, and explored the effects of flat tilted surfaces on depth estimation. In doing so, they added a

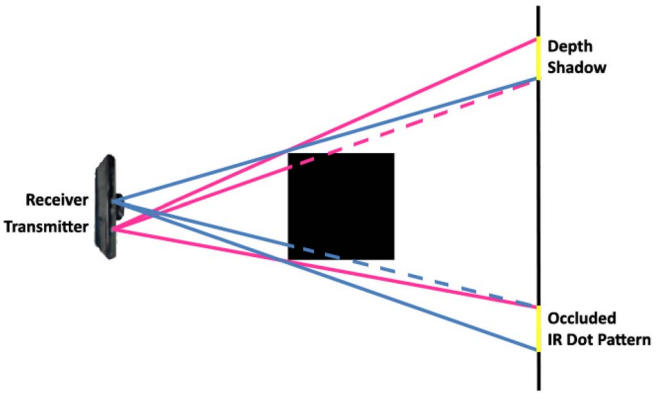


Fig. 4. A schematic is provided to explain the cause of IR dot occlusions and depth shadows. This mechanism also motivates the design for the simulated IR dot visibility detection, discussed later in Section IV-A2.

hyperbolic term to their quadratic axial and linear lateral models by fitting parameters to experimental data. This hyperbolic term accounts for an observed increase in measured variance with an increase in surface tilt angle and decrease in surface depth. Nguyen *et al.* [26] imprecisely related their models to the sensor's point spread function, which was characterized as the distribution of absolute errors from the distance of the observed edges to fitted lines. Instead, Mutto *et al.* [27] correctly explained that IR dots projected on excessively slanted surfaces are spread over more than two predownsampled pixels, where shifted dots characterized by various disparities within the correlation window can lead to a poor match from the correspondence algorithm. It should be noted that due to the nature of empirical models, the estimates of model coefficients are strongly dependent on the measured data sets, and errors from any hidden variables not accounted for will be mixed into the estimated parameter values. In the case of [26], factors such as surface material properties and ambient thermal conditions are implicitly kept constant since they are not modeled. Also, since depth data was collected on surfaces near the center of the optical axis, error due to lens distortion was not considered.

Choo *et al.* [28] presented another set of empirical models for axial and lateral error in recognition of nonlinear pincushion distortion that contributes to an increase in depth error with an increase in radial distance from the optical center. Here, the boundary of a grid of blocks was fitted to the depth data to construct estimates for ground truth images. It is noted that the observed error in depth images is due not only to unpredictable IR noise, but also to systematic quantization and mismatched correlation windows from measured sections of a distorted dot pattern. Choo *et al.* [28], however, did not account for changing environmental and surface properties, and assumed that varying distances between the object and background surfaces do not have an effect on the standard error of estimated object edges. Additionally, both Nguyen *et al.* [26] and Choo *et al.* [28] cannot provided bias errors since they do not have available ground truth, and are relegated to using the edge fits to the measured images.

It has been demonstrated by several reports that post-processing filters can reduce certain causes of error to improve the accuracy of depth image estimates. For instance, Tosic and Drewes [29] and Yang *et al.* [30] provided methods to inpaint nonmeasured depth values due to certain causes (e.g., specular and absorptive surfaces) by respectively utilizing Kinect IR and RGB images. Yu *et al.* [31] also provided a depth map repair method to fill in missing values due to IR image shadows, which are caused by IR dots obstructed from an object separated at a distance from its background. Shadowing and occlusion are depicted in Fig. 4, where the region of a 3-D scene (upper yellow section) imaged by the receiver (blue line-of-sight vector) cannot be reached by IR dots emanating from the transmitter (magenta line-of-sight vector). There may be benefit from predicting the size and shape of shadows, though, since they produce estimable systematic errors.

Unfortunately, all reported error models are directly formed from depth images and overlook the intermediate IR images that are used to generate them. As many researchers point out ([25], [27], [28], etc.), various sources of error (e.g., disparity quantization and missing depth due to shadows) are systemic, many of which are predictable given a CAD model of the scene. Additionally, it may be possible to formulate disparity mismatch error statistics as a function of the nonuniform distribution of IR dots and depth variation within correlation windows. There is also little focus placed on analyzing the effect of IR intensity variation due to dot splitting, spreading, and random speckle and IR detector noise. These effects contribute to unpredictable components of random depth error and are, therefore, an important factor to consider for adequate error model construction.

III. Proposed IR Intensity and Noise Models

In this section, we present a combination of physical and empirical models for the intensity of the received IR dots, which includes probability density functions for multiplicative speckle and IR detector noise. Since Kinect's subpixel refinement step relies on the intensities of dots splitting neighboring pixels to determine the best match from a similarity algorithm, random IR sources directly affect disparity estimation. These effects are studied later in this report, where we examine depth data sets of matte-painted planar walls parallel to the optical axis, i.e., when dot pattern distortion due to spreading is minimal. In order to construct the intensity models, we utilized the corresponding IR image streams of the flat, diffuse surfaces, which were recorded by a Kinect for Windows sensor positioned at various distances. A total of 17 distinct 100 frame data sets consist of surface depths between 800 and 4000 mm, separated by 200 mm. Since the ADC resolution is 10-bit on-chip [22], these receiver images consist of 1024 levels of quantized integers representing the detected IR intensities.

In the analysis of the data sets, we model the detector outputs Z as the expected dot intensity I corrupted by random speckle $\underset{\sim}{n}_I$ and additive noise $\underset{\sim}{n}$

$$Z = I + \underset{\sim}{n}_I + \underset{\sim}{n} \tag{2}$$

where $\underset{\sim}{n}_I$ is proportional to I. The random speckle is assumed to vary for the different surface depths, but remain constant

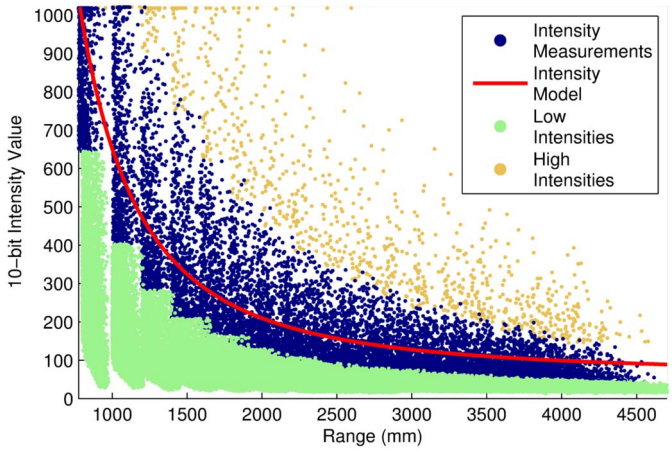


Fig. 5. The filtered intensity samples generated from unsaturated IR dots (blue dots) were used to fit the intensity model (red line), which follows an inverse square model for the distance between the sensor and the surface point.

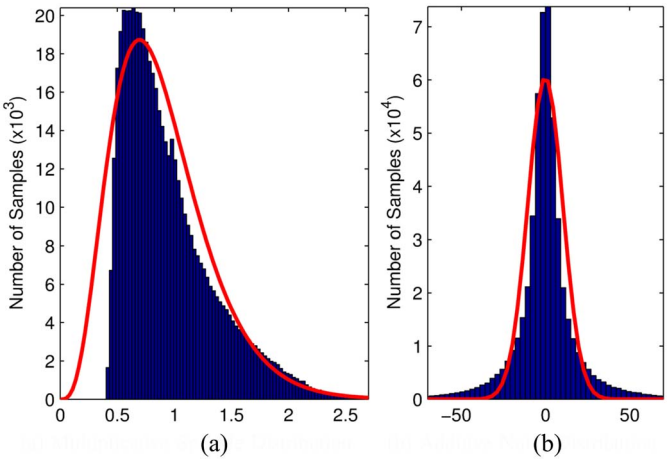


Fig. 6. (a) Multiplicative speckle distribution is unitless, and can be represented as a gamma distribution $\Gamma(4.54, 0.196)$. (b) Additive detector noise distribution can be represented as a normal distribution $\mathcal{N}(-0.126, 10.4)$, and has units of 10-bit intensity.

during the 30 Hz time samples. Therefore, detectors containing dots are modeled by

$$Z_{i,j,k,t} = \underset{\sim}{\gamma}_{i,j,k} \cdot I_{i,j,k} + \underset{\sim}{n}_{i,j,k,t} \tag{3}$$

where i and j are the pixel column and row ID, k is the depth, and t is the time sample. Here, $\underset{\sim}{\gamma}_{i,j,k}$ is the unitless, random intensity multiplicative term providing the random speckle, and $\underset{\sim}{n}_{i,j,k,t}$ represents random detector noise that does vary with time. Note, it is assumed that the speckle and detector noise variables are independent $\underset{\sim}{n} \perp\!\!\!\perp \underset{\sim}{\gamma}$.

The IR intensity samples $Z_{i,j,k,t}$ from the aggregate collection of all 17 data sets are scatter plotted in Fig. 5 as a function of range, i.e., the distance between the sensor and surface point. For each given depth trial, range varies by pixel angular values, therefore the bands in the figure represent intensities from different depth sets. Following Reichinger's binary dot pattern model, we filtered out the lower 88.9% of IR image intensities (green dots), separately for each depth. This is under the assumption that these pixels contained only ambient intensity and are not directly contributed by the dot pattern. We also filtered out the top 1% of the highest intensities (yellow dots), again separately for each depth, to avoid overfitting saturated values resulting from very bright IR dots.

As Zelevsky *et al.* [11] explained, the brightness of a detected dot depends on the range r between the illuminator and surface point, where the IR intensity I falloff rate follows the inverse square law, i.e., $I \propto 1/r^2$. Many additional factors influence the detected intensity [32]; however, Choe *et al.* [33] argued that the Kinect intensity model can be simplified to the albedo of the surface, global brightness, surface normal, and transmitter lighting direction by following the Lambertian bidirectional reflectance distribution function (BRDF) model. They also include an additional offset value I_a in their model, which accounts for a constant ambient intensity. This model was adopted by assuming constant surface reflectance and global brightness from the environment, and finding the least-squares fit of IR intensities to range, surface normal $\mathbf{n}$, and lighting direction $\mathbf{l}$. Thus

$$I = I_a + \frac{\alpha}{r^2}(\mathbf{n} \bullet \mathbf{l}) \tag{4}$$

where $I_a = 62.3$ units of intensity and $\alpha = 5.90 \times 10^8$ unit-mm^2 from the fitted model, which resulted in a 0.814 R^2 value. This high coefficient of determination implies that the variance in over 80% of the intensity data can be explained by (4). The remaining variance in data can be attributed to random speckle $\underset{\sim}{n}_I$ and detector noise $\underset{\sim}{n}$, which are assumed zero mean and independent for the fit. As Chow *et al.* [34] point out, indoor lighting conditions and object surface color have little effect on the detected IR intensity. Therefore, the constant ambient intensity can be attributed to thermal radiation in our experimental setup. Note, in order to provide highly accurate range estimates of the transmitted dots, a least-squares solution to the best-fit plane was performed over the 100 frame depth data sets for each IR stream. Dot ranges were then computed as the distance from the sensor to the refined plane estimate.

In order to retrieve relatively noise-free samples of the random speckle variable $\underset{\sim}{\gamma}$, time-averaged IR images were first determined. The speckle samples were subsequently generated by normalizing the average IR values by the corresponding predicted intensities, which were estimated using the fitted model in (4). Fig. 6(a) depicts a histogram of the resulting random speckle deviates that remain after we filtered out pixels that generally contain either no dots or saturated intensities, i.e., the lower 88.9% and upper 1% of pixels.

Laser power detected by a pixel is generally modeled as the sum of several exponentially distributed power (Rayleigh voltage) random variables [32], which results as a gamma distribution. We, therefore, fit a doubly truncated gamma to the deviates to recover the gamma distribution given by

$$f_{\underset{\sim}{\gamma}}(x; k, \theta) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} \mathrm{e}^{-\frac{x}{\theta}} \tag{5}$$

where the shape k and scale θ parameters were estimated to be 4.54 and 0.196, which produces a mean and standard deviation of $k\theta = 0.892$ and $\sqrt{k}\theta = 0.418$. The recovered gamma distribution is overlaid with the red line fit in Fig. 6(a). Note that these fitted parameters support the previous assumption that the mean value for $\underset{\sim}{n}_I$ is nearly zero.

Lastly, random IR detector noise deviates were generated by removing the average IR image separately from each depth data stream. A histogram of the resulting additive deviates $\underset{\sim}{n}$ is shown in Fig. 6(b), which follows a Gaussian distribution:

$$f_{\underset{\sim}{n}}(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \mathrm{e}^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{6}$$

where the fitted mean μ and standard deviation σ are -0.126 and 10.4 intensity units, respectively. The mean is less than one intensity unit and the standard deviation is expectedly small, which accounts for residual thermal noise present in the detector.

IV. Putting it All Together: Kinect IR and Depth Image Simulator

With the technical background and IR modeling presented in Sections II-A and III, we are now able to simulate Kinect IR and depth images of any scene by closely emulating the system and processes that construct the respective real images. In the following sections, we outline our models for the constructed Kinect simulator, which was coded and tested in MATLAB. We have also made our code publicly available on MATLAB Central's File Exchange Resource Center with an included demo, which takes in as input a transformed CAD model that defines the 3-D scene, and outputs the corresponding noisy IR and depth images. The sensor specifications are also tunable input arguments, though the default parameters such as the IR intensity and noise model values are taken from the literature. It should be understood that this simulation generates single image data of 3-D scenes, which closely resembles the invention presented in [11].

A. Simulating IR Images

The IR image simulator incorporates many key characteristics of the real IR transmitter/receiver system, such as the laser projection system specified by [17] and [18], and the CMOS sensor specified in the Aptina data sheet [22]. As mentioned earlier, the Kinect dot pattern uncovered by Reichinger [9] is also incorporated and is key to our implementation. This implies that the model represents an idealized Kinect system with a perfect factory construction. Therefore, there is no rotation in the projected IR dots, the epipolar lines are rectified, and there are no gaps nor overlap in the tiled pattern. For now, the effects of lens distortion deviating the rectilinear projection of dots are excluded, though this could be accounted for in a future iteration of the simulator software. Note, since Kinect's downsampled raw IR images are 640×480 pixel grids, the simulation pads/crops from the 633×495 FOV that the idealized dot pattern spans in order to preserve the final depth image resolution. Here, padding is achieved by adding a repeated portion of the simulated pattern to the leftmost and rightmost columns of the original dot pattern.

1) Transmitted IR Dot Pattern: In order to emulate the diffuser unit, line-of-sight vectors, or rays, are constructed from each pixel of the pattern that contains a dot [i.e., the white pixels in Fig. 2(d)]. These rays in effect represent the IR laser system that transmits the pattern onto a given scene. Since the transmitted dots have a physical cross-sectional area, the rays are divided into a grid of subrays with c_{sub} columns and r_{sub} rows. Note, it is necessary to construct at least eight columns of subrays to accommodate Kinect's subpixel disparity accuracy of (1/8)th of a pixel. For this report, the subray grid was set to $c_{\text{sub}} = 17$ columns and $r_{\text{sub}} = 7$ rows. Though these tuning parameters can be altered at the discretion of the user, we have determined from rigorous experiments that a lower subray column resolution adversely affects disparity refinement accuracy, whereas a higher row resolution is unnecessary and does not affect accuracy.

Next, let $\mathbf{p}_T = [u_T, v_T]^\top$ represent the coordinate of a subray measured in pixels, and referenced to the transmitter coordinate system. The end point of each subray $\mathbf{X}_T = \left[x_T, y_T, z_T\right]^\top$ is then constructed to intersect a flat wall at Kinect's maximum range $Z_{\max}$. These are computed by utilizing the lens equation and the horizontal f_x and vertical f_y focal lengths of the sensor

$$\mathbf{X}_T = Z_{\max} \cdot \left[\frac{u_T}{f_x}, \frac{v_T}{f_y}, 1\right]^\top + [-b, 0, 0]^\top. \tag{7}$$

The horizontal and vertical focal lengths are determined to be $f_x = 571.4$ and $f_y = 570.9$ pixels, respectively, using the known image size and FOVs. Since it is assumed that the sensor and receiver coordinate systems are collocated, the X-axis coordinates of each transmitted subray are shifted by the baseline distance b. A ray casting method [10] is then used to determine the 3-D location where each subray intersects the inputted CAD models. For each subray that does in fact intersects the CADs before reaching $Z_{\max}$, the algorithm returns an output fractional distance δ_T less than 1, whereas a δ_T of 1 represents a nonintersecting subray.

2) Received IR Dot Pattern: The locations of CAD model intersections are used to determine which parts of the simulated IR dots would be visible to the receiver. Here, receive subrays are constructed that emanate from the origin of the receiver coordinate system (0, 0, 0) and end at the CAD model intersections, i.e., $\mathbf{X}_R = \delta_T \cdot \mathbf{X}_T$. The ray casting algorithm is redeployed and if the intersection with the CAD models is closer than expected, the returned fractional distance δ_R is less than 1, indicating occlusion. For a visual depiction of this, refer to the lower transmitter ray (solid magenta line) and the intersecting receiver ray (dashed blue line) in Fig. 4. Only transmitted subrays that both intersect the CAD models and are not occluded contribute to the IR image. For each intersecting and unoccluded subdot, the 3-D end point is finally projected into the receiver pixel coordinate system via the lens equation to provide $\mathbf{p}_R$. It should be noted that the transmit and receive subrays require shifting to provide row and column indices with respect to the origin of the corresponding grids, which are both ordered from top to bottom and left to right.

Also, we assume that transmit dot sizes are equal to the receive pixel spacing, which is roughly true according to [11].

3) IR Detector Integration and Intensity: We assume the Kinect sensor follows the linear property of optical imaging systems, where each dot is imaged simultaneously and integrated independently to create the simulated IR image. As follows, each dot's subray is summed directly into the final grid, thereby circumventing the accessory 2×2 binning step. Again, each row of the transmitted dots line up with its coinciding row in the received image, therefore the subrays within an IR dot can only spread horizontally, and cannot spread vertically. Similarly, the depth of dots intersecting a flat surface orthogonal to the Z-axis determines the disparity shift, and thus the amount of horizontal pixel splitting. To illustrate this point, let an IR dot with a subray grid size of 17×7 intersect a flat, orthogonal wall. If the first five columns of subrays fall within one pixel, and the remaining 12 columns of subrays fall within the neighboring pixel, the two pixels would receive $5/17 = 29.4\%$ and $12/17 = 70.6\%$ of the full IR dot intensity [rightmost window of Fig. 7(a)]. This design permits faithful predictions of subpixel refinement performance, which is employed in Section IV-B2. In the case of a tilted surface, however, dots can be spread over more than two pixels, such as the rightmost window in Fig. 7(b). Here, a single dot is spread to three pixels when a surface is rotated $-80°$ about the Y-axis. Unlike the estimable flat wall projections, these imperfections and distortions in the measured dot pattern lead to systematically larger depth errors.

The analysis and generation of the empirical models presented in Section III are then used to simulate the IR intensity, speckle, and noise that effect the receive IR detectors. More specifically, a modified version of (3) and (4) is used to simulate 10-bit intensity values for each pixel Z that contains the sum of subrays (i, j) representing a dot's partial energy falling within it

$$Z = \underset{\sim}{\gamma} \frac{\alpha}{c_{\text{sub}} \cdot r_{\text{sub}}} \sum_{i,j} \frac{\mathbb{1}_{i,j} (\mathbf{n} \bullet \mathbf{l})_{i,j}}{r_{i,j}^2} + \underset{\sim}{n}. \tag{8}$$

Thus, the indicator function $\mathbb{1}_{i,j}$ takes on a value of 1 when subray (i, j) is visible within the pixel, and a value of 0 when it is occluded. Random speckle $\underset{\sim}{\gamma}$ is drawn from the gamma distribution in (5) and applied to each simulated dot, separately, whereas random detector noise $\underset{\sim}{n}$ is drawn from the Gaussian distribution in (6) and applied to each pixel, separately. Note that the constant ambient offset I_a from (4) is left out since it depends on the experimental setup. Also, $(\mathbf{n} \bullet \mathbf{l})_{i,j}$ and $r_{i,j}^2$ depend on $\mathbf{X}_R$ of the subray and the unit normal of the intersected CAD facet. Since Kinect quantizes the received IR images, intensity values Z are finally rounded to the nearest integer value.

The resulting IR image of a simulated block grid, similar to the experimental setup implemented in [28] to estimate edge error, is displayed in Fig. 8. Note the existence of shadows along the left edge of some blocks that obstruct sections of the background wall. The known pattern of "wavy" edges is also apparent due to the projection of the nonuniform distribution of dots on the surface of the scene. From this paper, this is believed to be the direct cause of the standard lateral error observed in many reports (including [26] and [28]), which is further explored in Section V.

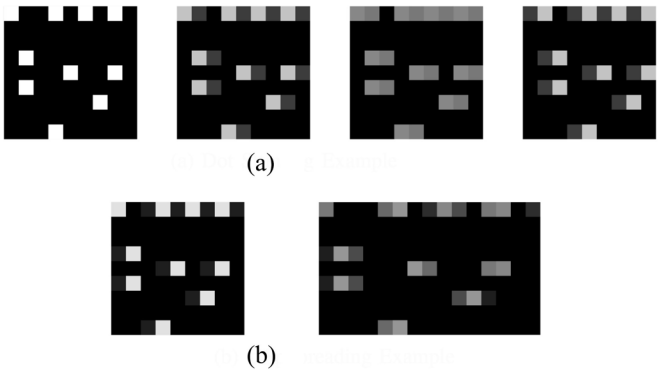


Fig. 7. The leftmost window in (a) depicts IR dots intersecting only one pixel from a flat wall projection, whereas the other windows depict IR dots splitting two pixels. The rightmost window of (b) shows an example of the dot pattern spread over multiple pixels from an 80° tilted wall projection.

B. Simulating Depth Images

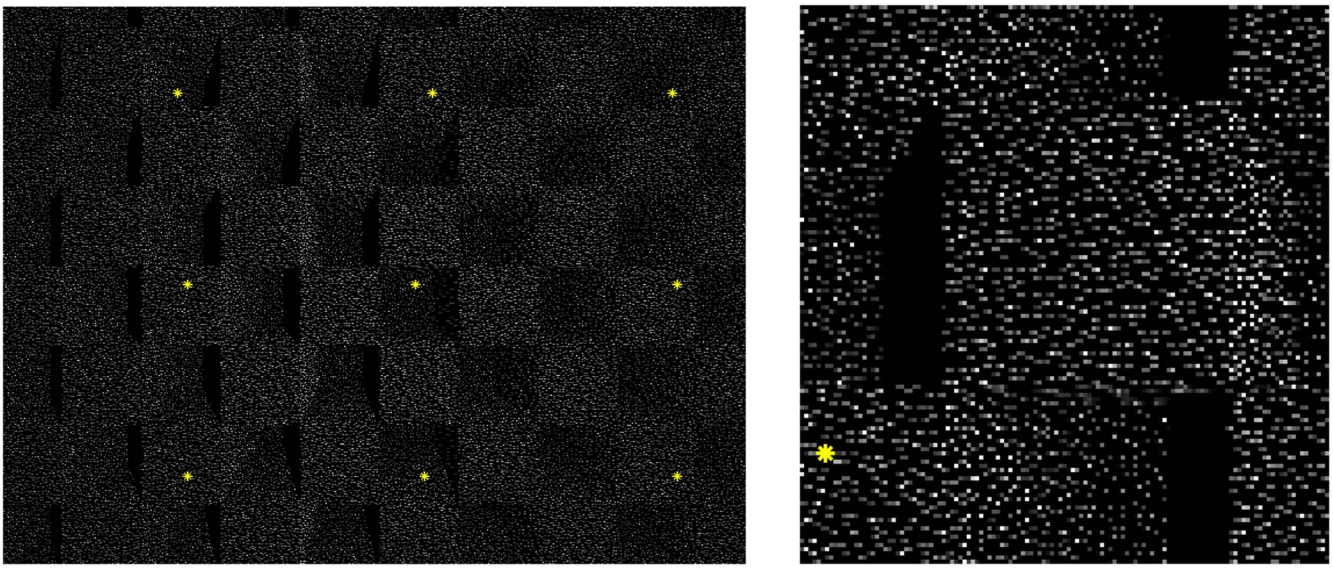
In this section, the methods for processing simulated IR images to generate depth images are provided. As mentioned previously, it is believed that Kinect employs a thresholding scheme that converts noisy IR images to binary images prior to the first disparity estimation step. Since the specifics of Kinect's filter is unknown, for practical purposes, we assume the filter implemented on the device's SoC works sufficiently well, and contributes little to no error in the initial disparity estimates. Therefore, our model utilizes postfiltered IR images for the correlation-based algorithm, where knowledge of pixels with unoccluded dots is given. It should be noted that while correlation-based disparity estimation utilizes filtered IR images, the subpixel refinement step processes the prefiltered, noisy IR images in both the real and the simulated Kinect systems.

1) Estimate Disparity With Correlation Window: As established in Section II-A, a local, pixel-based correlation algorithm is used to compare windows of the measured binary image to a series of reference binary images. The shift that provides the best match is determined by the reference window that maximizes the cross-covariance C among the set of N windows tested. Given the default operational range of depths falling between 800 and 4000 mm, the minimum and maximum allowed disparities are computed to be 53.57 and 10.71 pixels via (1). Therefore, the maximum disparity shift from any reference image is determined as $N = 45$ pixels.

Since the epipolar lines are coincident, the search for the best match between measured and reference image windows centered at (u_Z, v_Z) and $(u_{\text{ref}}, v_{\text{ref}})$ is constrained to be within the same row of pixels. This lateral shift translates to a change in disparity Δd_0, where $u_Z = u_{\text{ref}} + \Delta d_0$ and $v_Z = v_{\text{ref}}$. As follows, the initial disparity is estimated by:

$$\Delta d_0 = \underset{n \in N}{\arg\max}\, C_n \tag{9}$$

where Δd_0 is restricted to an integer value of pixels, and the cross-covariance score C_n of a reference window centered at



(a) (b)

Fig. 8. (a) Simulated IR image of block grid. (b) Magnified section of the IR image. The simulated IR image of a block grid emulates distinctive shadowing and "wavy" edges observed in real Kinect IR images. Note that IR dots projected on the surfaces of the blocks are brighter due to the inverse square law included in (8), and the bright dots (yellow stars) projected on the surfaces of the blocks and background represent different disparity shifts.

$(u_{\text{ref},n}, v_{\text{ref},n})$ is computed via

$$C_n = \sum_{(u,v)\in W_Z} \left[W_Z - \bar{W}_Z\right] \circ \left[W_{\text{ref},n} - \bar{W}_{\text{ref},n}\right]. \tag{10}$$

Here, W_Z and $W_{\text{ref},n}$ represent measured and reference binary image windows, and $\bar{W}$ denotes the average over the window. Due to the strong evidence supported by [19] and Table I, the correlation window is set to a default size of 9×9 pixels.

2) Subpixel Refinement: After the initial integer disparity is computed, the estimate is refined to provide higher depth image accuracy. Subpixel refinement estimates an interwindow fractional disparity Δd_s by determining where the IR dots split pixels within the window of an assumed constant depth, i.e., an orthogonal and flat surface segment. Unfortunately, the refinement method is not provided in the PrimeSense documentation, so it is unclear what matching metric is employed. Though there are many ways to measure the matching cost, the sum of absolute differences (SAD) is generally considered the most common pixel-based method due to its computational simplicity [23]. Therefore, the SAD cost between the measured and predicted image is utilized in our simulated model. And due to the analysis supported by [19], [24], and Fig. 3, the default interpolation factor is set to eight subpixel levels.

In contrast to Section IV-B1, which employs binary reference images, a series of non-noisy IR intensity reference images are utilized as a lookup table for the sub-pixel refinement step. More explicitly, reference images of a flat wall at various depths are preprocessed, which are determined using (1) and all integer disparity values between the minimum and maximum disparities. Given the IR intensities and dot pattern projection from the reference image corresponding to the initial integer disparity estimate Δd_0, an approximate prediction of all interwindow pixel split possibilities can then be supplied [see Fig. 7(a)]. The total pixel displacement is ultimately computed as the aggregate of the two disparity estimates, that is

$$\Delta d = \Delta d_0 + \Delta d_s. \tag{11}$$

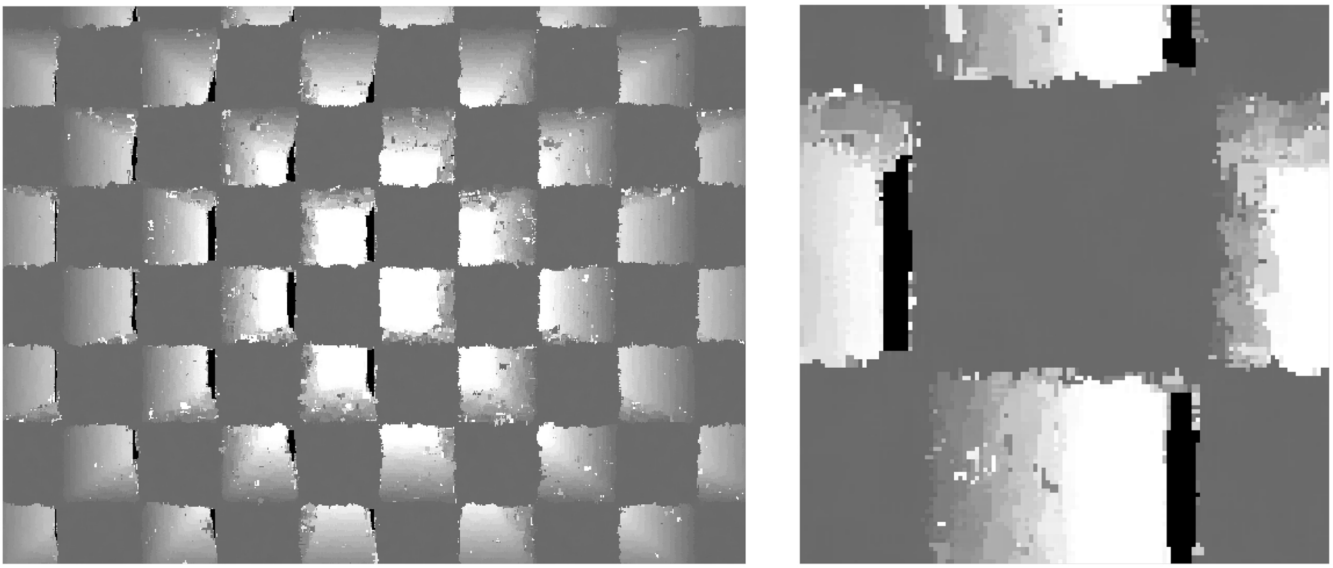
It is worth noting that the nonlinear spacing of reference images generated for the depth image simulator does in fact agree with the increase in spacing mentioned in [12].

3) Estimate Depth From Disparity: In order to compute depth from the total disparity estimate Δd, (1) was derived to estimate the change in depth with respect to a given reference image. Thus, a pixel's depth estimate is computed as

$$z = f_x \frac{b}{d_{\text{off}} + \Delta d} \tag{12}$$

where d_{off} represents the offset disparity of a reference image. Since the transmitter is physically positioned to the right of the receiver (Fig. 1), a positive Δd implies a pattern shift to the right, which translates to an object mapped closer to the sensor when compared to the reference depth. Alternatively, a negative Δd implies a shift to the left and an object mapped further away. It should be noted that by limiting total disparity estimation to an $(1/8)$th of a pixel with equal intervals, we are in essence quantizing allowable output disparity similar to the way Kinect quantizes depth images into 11-bit values (Fig. 3). The simulated depth error due to disparity quantization is, therefore, comparable to the real-valued depth errors from the Kinect system.

The resulting depth image of the simulated block grid is displayed in Fig. 9, which portrays key qualitative features seen in real depth images. For instance, edges of the blocks with straight and flat sides are imaged as the distinctly wavy edges. Artificial bridges and gaps are also introduced where block corners meet, which should otherwise intersect at a single point. These characteristics exist in Kinect because of imperfect matching estimates due to noise and significant depth



(a) (b)

Fig. 9. (a) Simulated depth image of block grid. (b) Magnified section of the depth image. The simulated depth image of a block grid propagates the shadows and wavy edges by processing the noisy IR image. Other distinctive characteristics of real Kinect depth images are also present, such as artificially induced bridges and gaps near the corners of each block.

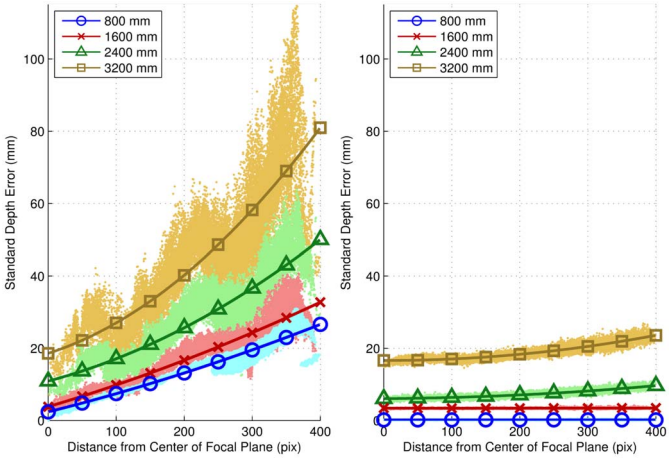
variation within a processing window. Moreover, depth shadows are observed when there are not any intersecting dots within a tested 9×9 correlation window, and as such, the algorithm cannot infer depth.

V. Results and Model Validation

In addition to a decisive qualitative comparison between simulated and real IR and depth images, quantitative comparisons of axial and lateral error are provided on three different simulated and experimental data sets. These data sets consist of 100 frame depth image streams from: 1) a series of flat walls parallel to the focal plane at various depths; 2) a series of flat walls with varying tilt angles; and 3) a flat-edged, square object placed at two distances from the background wall and at varying distances from the sensor. Errors obtained from the simulated scenes are also compared to the three models discussed in Section II-B to further support validation. Since their experimental results are highly dependent on the properties of the actual environment and sensor deployed, the resulting empirical models are not expected to perfectly align with our simulated results. However, the environment present in our experiments does coincide with the setup utilized to determine our speckle and detector noise distribution statistics in Section III. It is also important to note that while some experimental results may align with previously constructed error models, these models incorrectly assume homogeneity and only predict the average error statistics. Our model, on the other hand, captures these error statistics by replicating salient and inhomogeneous features of the system.

We first examined the standard deviation of depth error for the series of flat walls parallel to the focal plane. This metric is referred to as the standard depth error. The same best-fit planes constructed for the IR models in Section III were again used as the basis for depth ground truth. In Fig. 10, the standard depth errors are plotted separately for each pixel, which were computed using the 100 frame data sets. As seen here, the standard depth error as a function of radial distance increases at a faster rate in the experimental data. Since pincushion distortion stretches the dot pattern at further distances, it follows that Kinect's correspondence algorithm suffers when matching templates to the reference IR images. We, therefore, focus on the standard depth error for the collection of pixels closer to the optical center (closest 10%) for the subsequent quantitative comparisons to avoid having to account for lens distortion.

The standard depth error of the center pixels is plotted for each data set in Fig. 11 along with the Menna, Nguyen, and Choo models for flat wall depths between 800 and 4000 mm



(a) (b)

Fig. 10. The standard error in depth estimation (mm) as a function of radial distance (pix) is plotted for the (a) experimental and (b) simulated data sets of flat walls at various depths (mm). The experimental standard depth error increases faster with an increase in radial distance due to lens distortion.

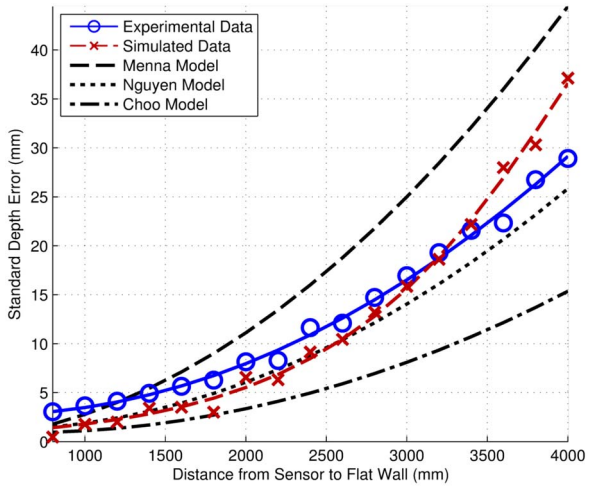


Fig. 11. The standard depth error (mm) as a function of flat wall depth (mm) is plotted for the experimental and simulated data sets. Three models are also plotted, where the Nguyen model complies most with both data sets.

at intervals of 200 mm. As seen in this figure, our simulated results closely comply with the experimental results for the range of depths tested, though standard error is perhaps slightly underestimated for closer flat walls. Additionally, the Nguyen model appears to fit most closely with the measured standard errors, whereas the Menna and Choo models noticeably overestimate and underestimate, respectively. Nonetheless, the overall quadratic axial error trend is present in each curve of the figure, which can certainly be attributed to the stereo system. Flat surfaces do not exhibit dot occlusion or dot spreading, however, so depth errors arise mainly when noisy IR intensities lead to imperfect matches from the SAD algorithm in the simulation. And since Chow and Lichti [35] demonstrate that error due to disparity quantization is minimal, the experimental standard errors near the center of the focal plane are decidedly attributed to subpixel refinement.

Next, the experimental and simulated standard depth errors of close pixels for each tilted wall data set are plotted in Fig. 12(a). Two data setups were constructed to fix the distance between the sensor and the center of the wall at 1000 and 1400 mm, and consist of the plane's normal varying between nearly orthogonal to nearly parallel to the optical axis. Ground truth was again acquired by finding the RMSE solution of the best-fit plane to the experimental depth streams. Since only the model provided by Nguyen *et al.* [26] accounts for surface tilt, their model is also plotted for comparison and validation. The figure shows that our experimental and simulated errors follow closely together along with the Nguyen model at tilt angles below 50°. The Nguyen model underestimates the experimental results at steeper angles, though, which is likely attributed to the difference in actual reflective properties (albedo) between their tested surface and ours. The simulated results, on the other hand, tend to overestimate standard depth errors at steep angles. The discrepancy can be explained with Figs. 12(b) and (c), which depicts errors plotted as a focal plane heat map collected for a surface at 80° and 1400 mm. Outlier values (yellow and blue pixels) peppered in the simulated images tend to skew the standard error, whereas the experimental depth errors tend to stay more consistent but contain significant regions with no result. As explained earlier, depth error becomes more appreciable for closer surfaces with steeper angles because dots are horizontally spread. We postulate that a region growing algorithm, believed to be employed in the Kinect system, is able to smooth poor matches from the correlation-based disparity estimation step (though not in all cases). Our simulator also occasionally finds poor matches

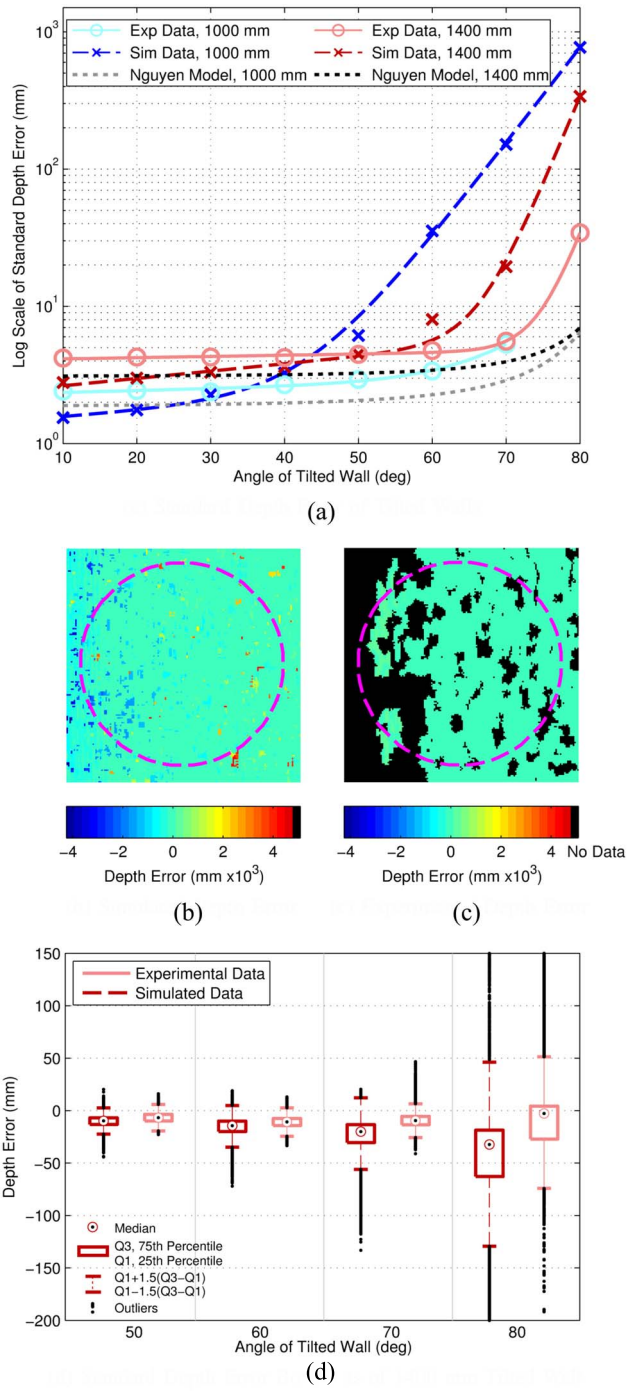
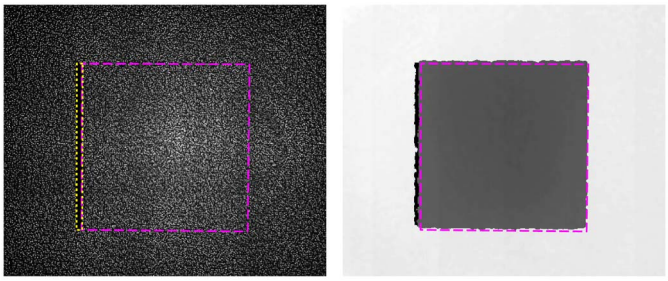


Fig. 12. The standard depth error (mm) as a function of tilted wall angle (deg) is plotted in (a) and compared to the Nguyen model. (b) and (c) show the simulated and experimental depth error heat maps of pixels close to the optical center (within the dashed magenta circles) for an 80° tilted wall, 1400 mm from the sensor. Tukey box plots are plotted in (d) for 50°–80° tilted walls, 1400 mm from the sensor.



(a) (b)

Fig. 13. (a) A novel method to fit the edges of a block is presented, which utilizes the IR shadow caused by the surface of the block obstructing a section of the background wall. (b) The resulting fit clearly shows a bias in the estimated edges when overlaid on the corresponding experimental depth image.

when the correlation peak is broad, but does not smooth pixels nor set them to a missing depth value. Upon examination of the Tukey box plots for the experimental and simulated depth errors from the 1400 mm tilted wall data sets in Fig. 12(d), it is clear that the histograms are similar, and primarily differ in the tails of the distributions represented by outlier values. Here, the tails in the experimental distributions are truncated due to smoothed or "no result" pixels that produced correlation peaks below a threshold.

Lastly, the lateral error component of Kinect is examined by determining the distances between the true and estimated edges of a square object placed at different distances from the wall and the sensor. We provide two metrics, bias of edge error and standard edge error, which respectively refer to the constant offset and standard deviation of the edge location errors. Here, the experimental IR data in Fig. 13(a) is used to obtain ground truth by first fitting planes to the block object and wall surfaces, and then by fitting a boundary to the object. To compute the box boundary, a thin boarder is first fit around the shadow (dashed yellow line) caused by the left edge of the object, and the right edge of the shadow box then initializes the left edge of the object box. The left corners of the object box combined with the knowledge of the object width and RMSE-fitted plane depth finally determine the remainder of the box boundary (dashed magenta line). This fitted boundary is overlaid on the corresponding depth image in Fig. 13(b).

Upon visual inspection of Fig. 13(b), it is clear that there is a shift between the boundary determined from the IR image (used as truth) and the estimated edges derived from the experimental depth image. Konolige *et al.* [19] suggested that this shift is the result of an IR camera to depth offset. After removing this offset from the experimental data sets, there remains a bias of error, as supported by the solid line plots in Figs. 14(a) and 15(a). More precisely, each experimental edge generally falls outside of the ground truth border (represented as a positive bias), which is consistent with the simulated data

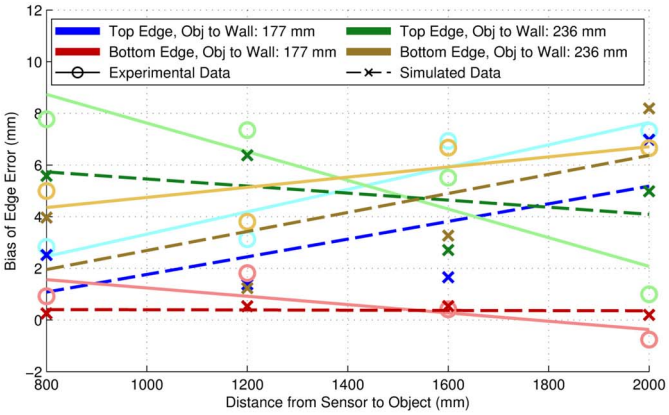


Fig. 14. (a) Bias of error (mm) for horizontal edges. (b) Standard error (mm) for horizontal edges.

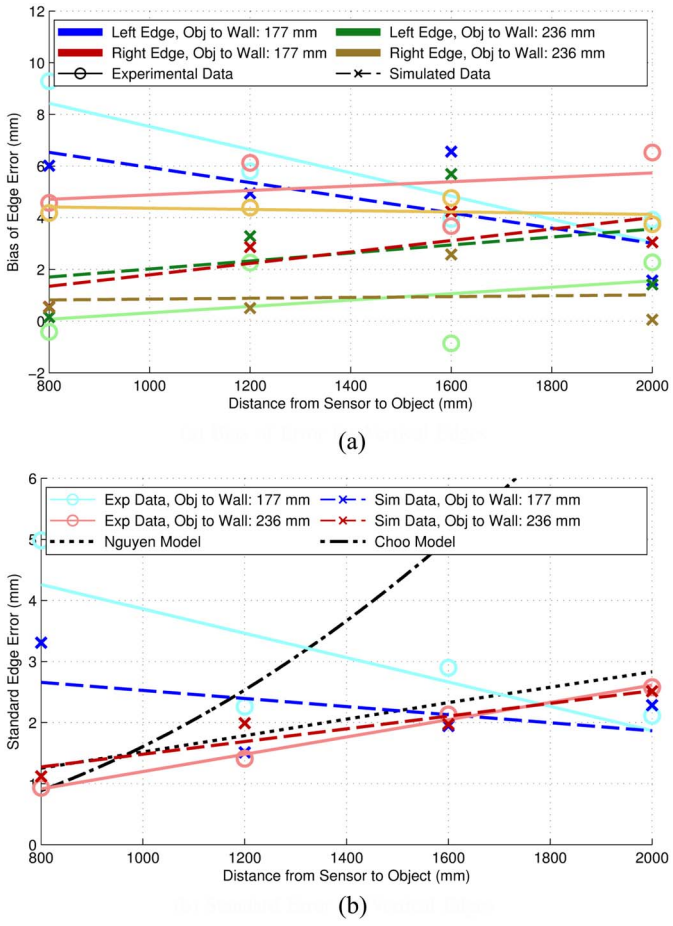


Fig. 15. (a) Bias of error (mm) for vertical edges. (b) Standard error (mm) for vertical edges.

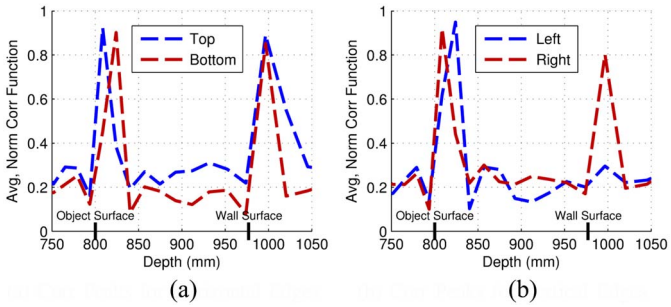


Fig. 16. The normalized and averaged correlation functions for (a) horizontal and (b) vertical simulated object edges are displayed. Note the two peaks for the top, bottom, and right edges, which cause depth ambiguity that result in the bias of edge error.

sets (dashed lines) in the respective figures. We, therefore, propose that the bias is the result of an imperfect stereo matching algorithm. This is easily explained at the left edge of the box when the correlation window operates in the shadow of the IR image and interpolates depth estimates. The top, bottom, and right edges have alternatively been determined to produce two correlation peaks corresponding to dot pattern segments on the object versus wall surfaces. This is demonstrated by the correlation functions displayed in Fig. 16. Here, the correlation values computed separately for each edge pixel were normalized and averaged for the simulated edges of the 800 mm sensor-to-object, 177 mm object-to-wall data set. The higher peak manifests semi-randomly, which results in a mix of disparity estimates that are biased toward either the object or wall surface at these boundaries.

Imperfect matches also contribute to the standard errors of the estimated horizontal [Fig. 14(b)] and vertical [Fig. 15(b)] edges. These standard edge errors give a quantitative measure of the wavy edges observed in Figs. 8 and 9. The Menna (not displayed) and Nguyen models tend to underestimate lateral error for closer objects, and the Choo model appears to overestimate lateral error for further objects. More importantly, these models do not account for varying object to background surface distances, which is a clear influence on edge estimation and lateral error.

VI. Applications and Future Work

The main intended application is for a useful Kinect IR and depth image simulation tool, but it can be further used to develop new Kinect error models from stochastic and systematic factors, as well as to refine and test object recognition, reconstruction, and pose estimation methods. The proposed system is also constructed to easily adopt other structured light systems with a known, spatially fixed pattern. For example, Intel's newly released RealSense 3-D camera transmits a random IR stripe pattern onto a scene, and appears to process depth maps similarly to Kinect. Additionally, the Asus Xtion 3-D sensor is based on the PrimeSense IR technology, which can be similarly deconstructed and applied to our developed model. The simulation may also be extended to other light projection systems, as long as the system parameters can be obtained or estimated.

Ideally, for any system, a faithful prediction model can be incorporated into an "optimal" estimator. As follows, the noise-free mode of our model can be used as part of a new class of estimators for object recognition and pose estimation. One could, for example, simulate a noise-free (i.e., expected) depth image of an object's CAD, and transform the depth image into a "predicted model point cloud" to align with a measurement point cloud derived from a real depth image. Point cloud methods, however, generally assume that the 3-D measurement errors of point pairs are independent and homogeneous, and are, therefore, suboptimal. Alternately, independence and homogeneity is a valid assumption for the speckle and detector noise in the intermediate and unprocessed IR images. Therefore, we plan to build an algorithm that predicts the raw IR images of an object's CAD model given a 3-D pose transformation, which should provide more optimal shape-matching results.

Since our model represents the ideal IR dot projection, a system needs to be devised that would calibrate the replicated dot pattern to any given real Kinect sensor. The most important addition would be to include a model for lens distortion in order to faithfully predict the IR dot pattern over the entire focal plane. This could possibly be done by implementing Brown's model [36] to estimate radial parameters, which has been previously adopted in [34] and [35]. Additionally, we can incorporate a treatment of the bright center dots as anchor points to determine the 3-D rotation required to align the 3×3 grids. The lens distortion and rotation parameters can then all be jointly estimated by optimizing the alignment of the center dots. The IR intensity and noise model can also be extended if external environmental properties and the albedo of each facet of the CAD model are provided. And finally, the initial disparity estimation step can be improved by incorporating the suggested local smoothing/region growing algorithm, which would account for the observed smaller depth errors on tilted surfaces and object edges.

VII. Conclusion

The widely used Microsoft Kinect sensor portrays depth error characteristics that are difficult to predict. Accordingly, numerous attempts to construct geometric, empirical, and statistical error models are largely too simplistic, ad-hoc, and imprecise. A main reason for the inadequacies is due to the fact that these models ignore the intermediate noisy IR images, which are processed by the way of Kinect's undisclosed Light Coding technology. We, therefore, provide a detailed model to produce noisy IR and depth images, which was motivated by an extensive study of the sensor's underlying mechanisms and performance characteristics, as well as our newly constructed empirical models for the intensity, speckle, and detector noise of the received IR dot pattern. Since our simulator accurately recreates salient image artifacts and has validated error statistics, researchers may use it as a tool to provide ground truthed data sets of any object/scene with accompanying CAD models. The proposed model can also be applied to a wide set of other applications that include constructing richer axial and lateral Kinect error models, improved object recognition and pose estimation methods, and developing a simulator for other depth sensors that employ a structured light system.

REFERENCES

[1] B. Ni, Y. Pei, P. Moulin, and S. Yan, "Multilevel depth and image fusion for human activity detection," *IEEE Trans. Cybern.*, vol. 43, no. 5, pp. 1383–1394, Oct. 2013.

[2] D. Tao, L. Jin, Z. Yang, and X. Li, "Rank preserving sparse learning for Kinect based scene classification," *IEEE Trans. Cybern.*, vol. 43, no. 5, pp. 1406–1417, Oct. 2013.

[3] N. A. Zainuddin, Y. M. Mustafah, Y. A. M. Shawgi, and N. K. A. M. Rashid, "Autonomous navigation of mobile robot using Kinect sensor," in *Proc. 5th IEEE Int. Conf. Comput. Commun. Eng. (ICCCE)*, Kuala Lumpur, Malaysia, Sep. 2014, pp. 28–31.

[4] S. Kim and J. Kim, "Occupancy mapping and surface reconstruction using local Gaussian processes with Kinect sensors," *IEEE Trans. Cybern.*, vol. 43, no. 5, pp. 1335–1346, Oct. 2013.

[5] O. Lopes, M. Reyes, S. Escalera, and J. Gonzalez, "Spherical blurred shape model for 3-D object and pose recognition: Quantitative analysis and HCI applications in smart environments," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2379–2390, Dec. 2014.

[6] M. J. Landau, P. A. Beling, and M. D. DeVore, "Efficacy of statistical model-based pose estimation of rigid objects with corresponding CAD models using commodity depth sensors," in *Proc. 40th IEEE Conf. Ind. Electron. Soc. (IECON)*, Dallas, TX, USA, Oct. 2014, pp. 3445–3451.

[7] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft Kinect sensor: A review," *IEEE Trans. Cybern.*, vol. 43, no. 5, pp. 1318–1334, Oct. 2013.

[8] R. A. Newcombe *et al.*, "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. 10th IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*, Basel, Switzerland, Oct. 2011, pp. 127–136.

[9] A. Reichinger. (Mar. 2011). *Kinect Pattern Uncovered*. [Online]. Available: https://azttm.wordpress.com/2011/04/03/kinect-pattern-uncovered/

[10] P. Terdiman. (Aug. 2002). *OPCODE*. [Online]. Available: http://www.codercorner.com/Opcode.htm

[11] Z. Zalevsky, A. Shpunt, A. Maizels, and J. Garcia, "Method and system for object reconstruction," Patent WO2 007 043 036 A1, Apr. 2007.

[12] J. Garcia and Z. Zalevsky, "Range mapping using speckle decorrelation," U.S. Patent US7 433 024 B2, Oct. 2008.

[13] A. Shpunt, "Optical designs for zero order reduction," U.S. Patent US20 090 185 274 A1, Jul. 2009.

[14] A. Shpunt, "Depth mapping using multi-beam illumination," U.S. Patent US20 100 020 078 A1, Jan. 2010.

[15] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli, "Depth mapping using projected patterns," U.S. Patent US20 100 118 123 A1, May 2010.

[16] A. Shpunt and B. Pesach, "Optical pattern projection," U.S. Patent US20 100 284 082 A1, Nov. 2010.

[17] *Teardown of the Microsoft Kinect*. (Dec. 2010). [Online]. Available: www.chipworks.com/en/technical-competitive-analysis/resources/blog/teardown-of-the-microsoft-kinect-focused-on-motion-capture

[18] *Hardware Info—OpenKinect*. (Feb. 2011). [Online]. Available: http://openkinect.org/wiki/Hardware_info

[19] K. Konolige, P. Mihelich, and A. Tsuda. (Dec. 2012). *Technical Description of Kinect Calibration*. [Online]. Available: http://wiki.ros.org/kinect_calibration/technical

[20] J. Smisek, M. Jancosek, and T. Pajdla, "3D with Kinect," in *Proc. 13th IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Barcelona, Spain, Nov. 2011, pp. 1154–1160.

[21] *Kinect for Windows SDK 1.8*. (Feb. 2012). [Online]. Available: http://msdn.microsoft.com/en-us/library/hh855347.aspx, accessed Dec. 2014.

[22] *Image Sensors—MT 9m001c12stm Data Sheet—Aptina Imaging*. (Jan. 2013). [Online]. Available: http://www.aptina.com/products/image_sensors/mt9m001c12stm/

[23] R. Szeliski, *Computer Vision: Algorithms and Applications*. London, U.K.: Springer, Sep. 2010.

[24] F. Menna, F. Remondino, R. Battisti, and E. Nocerino, "Geometric investigation of a gaming active device," in *Proc. SPIE Videometr. Range Imag. Appl. XI*, vol. 8085. Munich, Germany, Jun. 2011, pp. 80 850G–80 850G–15.

[25] T. Mallick, P. P. Das, and A. K. Majumdar, "Characterizations of noise in Kinect depth images: A review," *IEEE Sensors J.*, vol. 14, no. 6, pp. 1731–1740, Jun. 2014.

[26] C. V. Nguyen, S. Izadi, and D. Lovell, "Modeling Kinect sensor noise for improved 3D reconstruction and tracking," in *Proc. 2nd Int. Conf. 3D Imaging Model. Process. Visual. Transm. (3DIMPVT)*, Zürich, Switzerland, Oct. 2012, pp. 524–530.

[27] C. D. Mutto, P. Zanuttigh, and G. M. Cortelazzo, *Time-of-Flight Cameras and Microsoft Kinect*. New York, NY, USA: Springer, Mar. 2012.

[28] B. Choo, M. Landau, M. DeVore, and P. A. Beling, "Statistical analysis-based error models for the Microsoft Kinect depth sensor," *Sensors*, vol. 14, no. 9, pp. 17430–17450, Sep. 2014.

[29] I. Tosic and S. Drewes, "Learning joint intensity-depth sparse representations," *IEEE Trans. Image Process.*, vol. 23, no. 5, pp. 2122–2132, May 2014.

[30] J. Yang, X. Ye, K. Li, C. Hou, and Y. Wang, "Color-guided depth recovery from RGB-D data using an adaptive autoregressive model," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3443–3458, Aug. 2014.

[31] Y. Yu, Y. Song, Y. Zhang, and S. Wen, "A shadow repair approach for Kinect depth maps," in *Proc. 11th Asian Conf. Comput. Vis. (ACCV)*, Daejeon, Korea, 2013, pp. 615–626.

[32] J. W. Goodman, *Speckle Phenomena in Optics: Theory and Applications*. Englewood, CO, USA: Roberts and Company, Nov. 2010.

[33] G. Choe, J. Park, Y.-W. Tai, and I. S. Kweon, "Exploiting shading cues in Kinect IR images for geometry refinement," in *Proc. 27th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Columbus, OH, USA, Jun. 2014, pp. 3922–3929.

[34] J. C. Chow, K. D. Ang, D. D. Lichti, and W. F. Teskey, "Performance analysis of a low-cost triangulation-based 3D camera: Microsoft Kinect system," in *Proc. Int. Archives Photogramm. Remote Sensing Spatial Inf. Sci.*, vol. 39B5. Melbourne, VIC, Australia, Jul. 2012, pp. 175–180.

[35] J. C. K. Chow and D. D. Lichti, "Photogrammetric bundle adjustment with self-calibration of the primesense 3D camera technology: Microsoft Kinect," *IEEE Access*, vol. 1, pp. 465–474, Jul. 2013.

[36] D. C. Brown, "Close-range camera calibration," *Photogramm. Eng.*, vol. 37, no. 8, pp. 855–866, Jan. 1971.

Michael J. Landau (M'09) received the B.E. degree from the Department of Biomedical Engineering, Vanderbilt University, Nashville, TN, USA, in 2008, and the M.S. degree from the Department of Electrical and Computer Engineering, University of Virginia (UVa), Charlottesville, VA, USA, in 2011, with an emphasis on 2-D/3-D detection and tracking of cells with multiphoton microscopy, as well as ground targets with electro-optical/infrared (EO/IR) sensors. He is currently a Ph.D. candidate in the Department of Systems and Information Engineering, UVa. His current research interests include 3-D object recognition and pose estimation using depth sensors for manufacturing and military applications.

Benjamin Y. Choo (M'06) received the B.S. and M.S. degrees from the Department of Electrical Engineering, Yonsei University, Seoul, Korea, in 2005 and 2007, respectively, and the M.E. degree from the Department of Electrical and Computer Engineering, University of Virginia (UVa), Charlottesville, VA, USA, in 2012. He is currently in the Ph.D. program in the Department of Systems and Information Engineering, UVa.

Peter A. Beling (M'01) received the Ph.D. degree in operations research from the University of California at Berkeley, Berkeley, CA, USA, in 1991.

He is an Associate Professor with the Department of Systems and Information Engineering, University of Virginia (UVa), Charlottesville, VA, USA. He has held positions at the Center for Naval Analyses, Arlington, VA, USA, and the IBM Almaden Research Center, San Jose, CA, USA. He is active in the UVa site of the Broadband Wireless Applications Center, which is an Industry-University Cooperative Research Center sponsored by the National Science Foundation. His current research interests include decision-making in complex systems, with an emphasis on adaptive decision support systems and on model-based approaches to system-of-systems design and assessment. His research has found applications in a variety of domains, including mission-focused cybersecurity, reconnaissance and surveillance, prognostics and diagnostics systems in manufacturing, education and training, and financial decision-making.