Development of a Measurement Noise Model for Kinect-style Sensors

Lucas Chavez[1], Cláudio dos S. Fernandes[2], James Milligan[3],
Mario F. M. Campos[2], Luiz Chaimowicz[2], Ron Lumia[1], Rafael Fierro[1], and M. Ani Hsieh[3]

***Abstract*— Kinect-style RGB-D sensors have provided an economical depth sensing technology. However, measurements made with these sensors contain significant noise. This work aims to develop a measurement noise model for the depth values acquired from Kinect-style sensors using a statistical analysis of a large real-world data set. We present the methods used for collecting a data set as well as a summary of the statistical analysis performed to obtain a sensor model using a Kinect sensor. Our generated model shows that uncertainty is strongly influenced by distance and weakly influenced by incident angle. A verification of our model was performed using a second Kinect and an ASUS Xtion PRO. The results show our model would be a good approximation for other Kinect-style RGB-D sensors, but that a model may need to be generated for a specific sensor to obtain higher accuracy. Additionally, our implementation is freely available online.**

I. Introduction

Autonomous robots must be able to operate in the real world, perceiving the environment using their sensors and acting accordingly. In order to accomplish this, they must deal with the uncertainty that is inherent to real environments. As discussed by Thrun et al. in [1], there are several factors that contribute to a robot's uncertainty: environments are unpredictable and partially observable; sensors and actuators are noisy and faulty; and models of the environment and robot are approximated. To cope with various sources of uncertainty, probabilistic methods have been successfully used in several robotic applications such as localization, mapping and decision making.

One of the key components of probabilistic methods is a good sensor model. This model represents the process by which the sensor measurements are generated. A good sensor model can adequately explain the noise that is inherent in the observed measurements. Several sensors, and their respective sensor models, have been studied in the literature, ranging from classical odometry sensors [2], cameras [3] [4], lasers [5], to Time of Flight (ToF) cameras [6].

In the last couple of years a novel sensor gained attention in the robotics community. The comercialization of the Kinect™ by Microsoft brought with it an inexpensive depth sensor that uses an active range system to generate a depth map of a given environment [7]. The Kinect-style sensors provide RGB-D images which include both visual (RGB) and depth (D) values at 30 frames per second and 640×480 resolution. Several works have taken advantage of this sensor technology in scenarios such as environmental mapping [11], 3D reconstruction [8], gesture recognition [9], and altitude control of aerial vehicles [10].

The objective of this paper is to develop a measurement noise model for Kinect-style sensors. Differently from previous works, which have created ad hoc models for these types of sensors [8] [12], we performed a statistical analysis on a large data set to provide a value for uncertainty as a function of distance and angle of incidence of a measurement.

The rest of the paper is laid out in the following manner: Section II presents previous works related to the generation and verification of measurement noise models for depth sensors, especially those related to Kinect-style sensors. Section III discusses the approach used for generating the model, including both the theory behind the methods used as well as how they were implemented. Next, Section IV presents the results of our methodology and gives the model generated for the sensor. In this section, we also validate our model with a second Kinect and an ASUS Xtion PRO. Finally, Section V summarizes our results and the usefulness of our model.

II. Related Works

One of the major drivers behind sensor noise modeling is the problem of Simultaneous Localization and Mapping (SLAM). The most popular and well-used solutions to this problem have a Kalman Filter based framework. This field of work began in 1987 with Smith et al. [13] and continues today as in [14] and [5]. All these methods have some core requirements as described in [15]. One such core requirement is a measurement model by which to describe the measurement uncertainty. This is typically modeled by the Gaussian measurement equation

$$z(k) = h_k(x_k, m) + v_k,$$

where the geometry of the measurement is described by $h_k()$ as a function of pose x_k and a set of landmarks m; and v_k is a zero mean Gaussian additive noise. The parameters chosen for the distribution of v_k describe the measurement uncertainty and, in most applications, they are empirically estimated. In addition, the uncertainty is usually modeled as a constant for simplicity, which may be an incorrect assumption.

There have been works focused on creating a more complex sensor model from experimental data. The works which are most relevant to this paper create sensor models for ToF

[1] Mechanical Engineering Department and Electrical & Computer Engineering Department, University of New Mexico, Albuquerque, NM, USA {lucasc,lumia}@unm.edu, rfierro@ece.unm.edu
[2] Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brazil {csantos,mario,chaimo}@dcc.ufmg.br
[3] Mechanical Engineering Department, Drexel University, Philadelphia, PA, USA {milligan.james,mhsieh1}@drexel.edu

cameras. In Kahlmann et al. [16] the effects of distance and change of temperature on the measurements are looked at. The authors found that a planar surface provided the a good test surface for calibration purposes and generated a Fixed Pattern Noise model. The work of Chiabrando et al. [17] performed a similar calibration of a ToF but also considered the effect of measurement incident angle.

There have been two major works with Kinect-style RGB-D sensors in which an ad hoc sensor model was created. In Fallon et al. [12] they define a likelihood function in which σ_d is used to describe the measurement uncertainty. Their resulting model evaluates measurements that have larger depth values with higher variance. The next work by Newcombe et al. [8] is a well-known application of the Kinect sensor. They modeled the uncertainty as a function of distance and incident angle. In their work the model is presented as

$$W_{R_k} \propto \frac{\cos(\theta)}{R_k(x)},$$

where θ is the angle of incidence and $R_k(x)$ is the depth measurement. They use W_{R_k} as a way describe the uncertainty in the measurement.

The Kinect-style sensor model that we present here is based on thorough observation of real world data, rather than ad hoc models typically found in the literature for this class of sensors.

III. Approach

We determine a measurement noise model for the Kinect sensor through statistical analysis of experimental data. We have designed our model to be compliant with similar models presented in works such as [18] and the well-known [1] by Thrun et al. One thing to note is that these works aim to build a *sensor model* which takes into account four general types of measurement errors: measurement noise, unexpected objects, failure to detect objects, and random unexplained noise. In this work we are focus on modeling the measurement noise component of the sensor model. In [1] the measurement noise component is named p_{hit} and formally defined it as:

$$p_{hit}(z_t^k|x_t,m) = \eta \ \left(2\pi\sigma_{hit}^2\right)^{-1/2} \exp\left(-\frac{1}{2}\frac{z_t^k - z_t^{k*}}{\sigma_{hit}^2}\right)$$

Using this definition, the measurement noise is modeled as a narrow Gaussian centered about z_t^{k*} and with standard deviation of σ_{hit}. The Probability Density Function (PDF) is normalized by η and is limited to the range $0 \leq z_t^k \leq z_{max}$ where z_{max} is the maximum sensor range. The "ground truth" sensor reading is denoted by z_t^{k*}. This value can be found from an environmental map m and the robot pose x_t via ray tracing. The actual sensor reading is denoted by z_t^k. The PDF is defined using the difference between the actual sensor reading and the ground truth. Also, σ_{hit} is a constant, intrinsic noise parameter which, in practice, is often set by hand. In [1] a methodology to learn σ_{hit} through experimentation is described.

Our work determines σ_{hit} for the Kinect sensor from a large experimental data set. There are two fundamental issues to consider when determining this parameter:

- First, σ_{hit} is not simply a constant value. It has been observed [12] and [8] that the quality of measurements from the Kinect is dependent on the measurement distance d and also the angle of incidence α. Therefore, we modeled σ_{hit} as a function $\sigma_{hit}(\alpha,d)$.
- Second, when ground truth is determined with an environmental map m and robot pose x_t, the data set is not guaranteed to contain a thorough sampling of α and d combinations. Therefore, we developed an economical method to obtain a data set with a thorough sampling.

The solutions that we present for approaching these fundamental issues is what sets our work apart and makes it of value to the robotics community.

The remainder of this section is divided into three subsections. First, in the Data Acquisition subsection, we outline our technique for acquiring our data set. Next, in the Theoretical Discussion subsection, we will have a detailed look at some of the crucial aspects of our method. Finally, will be the Implementation subsection where we will explain the pseudocode of the procedure we used for producing σ_{hit} as a function of α and d.

A. Data Acquisition

Our data set was obtained by viewing a large flat wall from several different sensor *poses* with the Kinect. For each *pose*, 30 point cloud *shots* were acquired. Each point cloud consists of a set of 3D coordinates such that $p = \langle x,y,z\rangle \in \mathbb{R}^3$. Figure 1 illustrates the data acquisition process. Each *pose* can be expressed by (L,θ). Since our method is based on the error of each point p to the wall and the position of the wall is estimated with the point cloud, we do not require accurate positioning of the sensor for each *pose*. The chosen *poses* can be described by the sets

$$\{1, 1.5, \ldots, 5\text{m}\} \times \{0, 15, \ldots, 75°\} \cup \{5.5, 6, \ldots, 10\text{m}\} \times \{0, 15°\}.$$

We took data from 74 different *poses* such that the distribution of points in the (α,d) space covers most of the region defined by $\{\alpha \in [0°, 90°],\ d \in [0\text{m}, 10\text{m}]\}$.

We also obtained a validation data set using a second Kinect sensor and a ASUS Xtion PRO. For each sensor we chose 4 *poses* which can be described by $\{2.25, 3.75m\} \times \{22.5, 52.5°\}$. These *poses* were chosen to be well spaced in the set of *poses* from our large data set.

Special care was taken during the data acquisition process in order to obtain our data set. We made sure that environmental factors such as lighting and temperature were typical of an office environment and did not vary significantly. Also, we post processed the data in order to remove small portions of point clouds which were not measurements of the wall, such as the floor. Finally, the 30 *shots* acquired at each *pose* were captured with the sensor completely still.

B. Theoretical Discussion

Let us investigate some of the crucial theoretical aspects of our methodology before moving on. To begin, we will

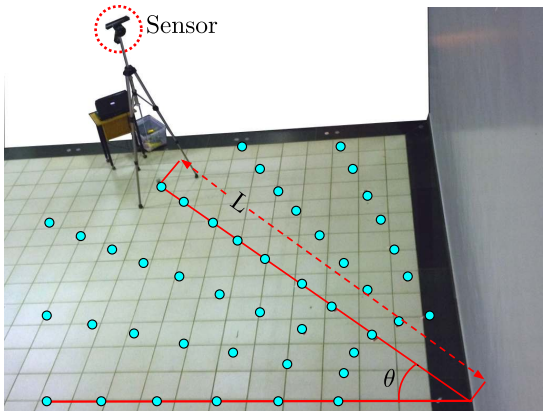

Fig. 1: Our experimental set-up. The Kinect is viewing the wall from a particular sensor *pose* defined by (L,θ). The spatial distribution of sensor *poses* for a subset of all *poses* are marked with cyan dots.

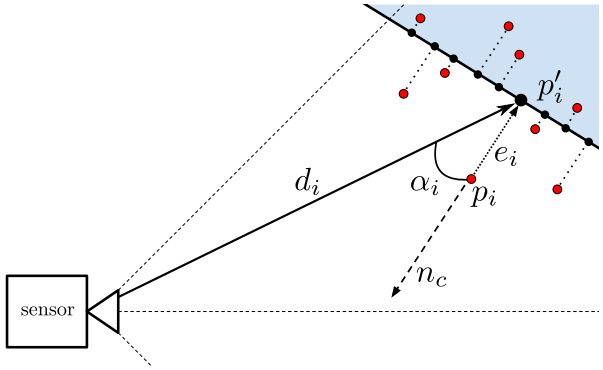

Fig. 2: Diagram showing quantities calculated for each point p_i. The blue shaded area represents our estimate of the wall. Red dots represents points from the point cloud. Here we see for a particular point p_i, which has measured the wall W_c with normal n_c, we can calculate p'_i, α_i, d_i, and e_i.

define an initial framework of our data acquisition process. First let us define the indices used in our framework: a is an index of all points in a single point cloud, b is an index of all *shots* for a single *pose*, and c is the index of all *poses*, such that:

$$a \in \{1,2,\ldots,n_a\};\ b \in \{1,2,\ldots,n_b\};\ c \in \{1,2,\ldots,n_c\}.$$

We will now define our estimate of the "ground truth". For each *pose* we define a wall by finding the best fit plane of all *shots* from that particular *pose*. The wall is defined by a point on the plane $\overline{p}_c$ and a unitary directional vector n_c by $W_c = [n_c, \overline{p}_c]$, where $\overline{p}_c$ is the centroid of all the *shots*

$$\overline{p}_c = \frac{1}{n_a n_b} \sum_{b=1}^{n_b} \sum_{a=1}^{n_b} p_{ab} \tag{1}$$

and n_c is the normal of the best fit plane found by using the centroid value and running a Principal Component Analysis (PCA) analysis for all points.

Let $i \in \{1,2,\ldots,n_a n_b n_c\}$ index all points from the entire data set. Then given a point p_i, we define the projected point p'_i, incident angle α_i, distance d_i, and error e_i.

$$e_i = (p_i - \overline{p}_c)\cdot n_c, \tag{2}$$

$$p'_i = p_i - n_c e_i, \tag{3}$$

$$\alpha_i = \cos^{-1}\left(\frac{-p'_i}{\|p'_i\|_2}\cdot n_c\right), \tag{4}$$

$$d_i = \|p'_i\|_2. \tag{5}$$

It is important to remember that, for each pose, the estimate of the wall is obtained from 30 point cloud *shots* and that this estimate uses points with a large range of α and d values.

Figure 2 illustrates the quantities we have calculated for each point. Using (2-5) the points can be represented in a new space described by α and d, which we will denote X. The space is defined such that $x_i \in X\ \forall i$ where $x_i = \langle \alpha_i, d_i \rangle \in \mathbb{R}^2$. The transformation of the points into this new space X is a central aspect of our methodology. After this transformation, each point still has an associated error e_i. It is important to remember, the fundamental assumption we have behind our analysis is that the variance of e_i values is dependent on α and d, so this new representation in X space will allow us to characterize this dependency.

In order to have a systematic way to characterize the dependance of the variance of e_i in the space X, we have defined a large set of neighborhoods N_j which has an index $j \in [1,2,\ldots,n_j]$ where n_j is the number of total neighborhoods. Each neighborhood N_j contains a set of points which have similar α and d values. We define a grid of locations $h_j \in X$ which are the centers of N_j. Then each N_j is defined as

$$N_j(h_j, r) \stackrel{\text{def}}{=} \{x_i \in X \ \mid\ \|(x_i - h_j)\circ(1,s)\| \le r\} \tag{6}$$

where r is the radius of each N_j. By adjusting r, we are adjusting the size of the N_j in the X space which lets us control the similarity of the points in N_j. We can also control the scaling on the d axis with s. The effect of s can be thought of as a transformation that scales an ellipse N_j into a circle. In Figure 3 we can visualize a small group of neighborhoods in the X space. The figure allows us to see how a set of points are grouped into a single neighborhood. Also, we can see the values we chose for the spacing between the neighborhoods and the values for defining the radius and scaling. Using this spacing, h_j becomes a 61×81 grid of points and $n_j = 4{,}941$.

We can now calculate the standard deviation of the error values e_i of the points in each N_j by using the standard deviation equation of (7) with μ defined as the mean.

$$\sigma_j = \sqrt{\frac{1}{n_k} \sum_{k=1}^{n_k} (e_k - \mu)^2} \tag{7}$$

Here $k \in [1,2,\ldots,n_k]$ is an index of all points contained within N_j. If an N_j has a $k < 500$, we do not consider it as part of our model generation.

In our calculations of σ_j we assume that the errors are

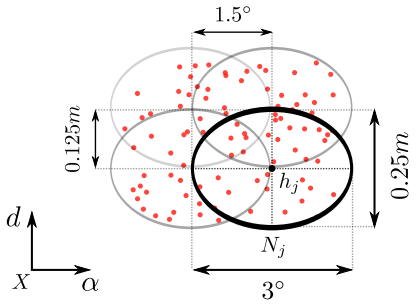

Fig. 3: Visualization of a small set of neighborhoods N_j and points x_i (colored in red) in the X space. We can see a particular N_j with center h_j. N_j is defined by all points within its perimeter.

drawn from a Gaussian process by $[e_1, e_2, \ldots, e_k] \sim \mathcal{N}(\mu, \sigma^2)$. Our goal is to obtain an estimate of the variance for the samples which we will denote $\hat{\sigma}^2$. Given that the position of the wall W_c is determined by the centroid $\overline{p}_c$, it can be shown that the distribution of all e_i will be centered around 0, i.e. for all points $\mu = 0$. For the sake of simplicity, we assume that this is also the behavior for each individual neighborhood N_j, assigning $\mu_j = 0$ to them. Below we further explain our reasoning behind this assumption.

We are interested in finding the best estimate of the sample variance $\hat{\sigma}^2$, so let us define the estimators of $\hat{\sigma}^2$ as

$$T_1 = \frac{1}{n_k}\sum_k (e_k - 0)^2 \tag{8}$$

$$T_2 = \frac{1}{n_k - 1}\sum_k (e_k - \overline{e}_k)^2 \ . \tag{9}$$

We can then compare these estimators by calculating the ratio of the mean squared error (MSE).

$$\frac{MSE_1}{MSE_2} = \frac{E(T_1 - \sigma^2)}{E(T_2 - \sigma^2)} = \frac{bias(T_1) + Var(T_1)}{bias(T_2) + Var(T_2)} = \frac{\frac{\sigma^4 Var(\chi^2_k)}{k^2}}{\frac{\sigma^4 Var(\chi^2_{k-1})}{(k-1)^2}} = \frac{k-1}{k} < 1. \tag{10}$$

From the ratio of the MSE values we can say that setting $\mu_j = 0$ gives a better estimate of σ_j. In addition, we have calculated $\overline{e}_k$ for each N_j, as can be seen in Figure 6. Since the overall mean values are distributed close to zero, we are confident that the zero mean assumption does not compromise the quality of our model, while giving us the benefits of a simpler model.

C. Implementation

Our method is summarized in Algorithm 1. It takes about 6 hours to process our complete dataset with a MatLab code, and about 2 hours to process it with the equivalent C++ version. Our code is freely available in a public repository[1].

[1]More info about the repository can be reached through http://www.verlab.dcc.ufmg.br/projetos/kinect-measurement-noise-model

Algorithm 1 $\sigma(\alpha, d) = GenerateModel(p_{abc})$

```
% Fill all neighborhoods N_j with its points
for c = 1 : n_c                    % each position c
  find W_c = [n_c, p̄_c]            % equation of the best-fit plane
  for b = 1 : n_b                  % each shot b
    for i = 1 : n_a n_b n_c        % all points
      find p'_i, α_i, d_i, and e_i
      x_i = [α_i, d_i]
      for j = 1 : n_j              % all neighborhoods
        if ||(x_i − h_j) ∘ (1, s)|| ≤ r
          x_i belongs to N_j

% Calculate the variance σ_j for each N_j
for j = 1 : n_j                    % all neighborhoods
  σ_j = sqrt((1/N) Σ_k (e_k − μ)^2)

σ(α, d) = fitPolySurface(σ_j, h_j)
```

IV. Results

With the captured data, it was possible to fill 72.9% of the original neighborhoods with, at least, a threshold of 500 points. The distribution of number of points across the X space is illustrated by Figure 5.

We performed a quantile-quantile regression with the data shown in the histogram from Figure 4 with a gaussian distribution, and obtained a coefficient of determination of 0.805, which validates the hypothesis that the error e_i can be approximated by a Gaussian random variable with zero mean. Figure 6 shows the average of the error calculated from the samples at each neighborhood N_j. Since most of its values are dispersed close to zero, we can say that the approximation of a zero-centered distribution for each neighborhood does not compromise our model, whilst making it simpler for real applications.

Next we will show the results of our model generation. In Figure 7 we can see the variance σ_j in each neighborhood N_j and observe that σ_j increases with distance, which is a trend we expected. One important observation is that, according to our model, the angle α does not play a major role in determining the variation σ_j. In Figure 8 we can see a scatter plot showing our generated polynomial surface $\sigma(\alpha, d)$; σ_j obtained from our data set; and σ_j obtained from our validation set. The surface represents the best fit polynomial surface through the original estimations. The green dots represent the original estimations; the red ones were acquired from the XtionPRO, while the black points were computed from a different Kinect sensor.

We asses the ability of our model to describe our data by the coefficient of determination. This allows us to determine the percentage of points whose variation are described by $\sigma(\alpha, d)$. In Table I we can see the value of this coefficient for our large data set and also our validation set. Our results show that our model could be used for other Kinect device with a reasonable confidence, but that a new model would be needed for the Xtion, should the application require a high

accuracy model.

TABLE I: Coefficient of determination.

Original Kinect	Another Kinect	ASUS XtionPRO
90.0%	74.7%	50.7%

The resulting measurement noise model from our analysis is given by (11) and the list of coefficients are in Table II.

$$\sigma(d,\alpha) = A + B\alpha + Cd + D\alpha^2 + E\alpha d + Fd^2 + G\alpha^3 + H\alpha^2 d + I\alpha d^2 + Jd^3 \quad (11)$$

TABLE II: Polynomial coefficients for the surface $\sigma(d,\alpha)$.

A	0.0125	F	0.0037
B	-6.0904×10^{-4}	G	3.4986×10^{-8}
C	-0.0061	H	-3.9492×10^{-6}
D	8.0999×10^{-6}	I	-2.7408×10^{-6}
E	1.5757×10^{-4}	J	-1.1158×10^{-4}

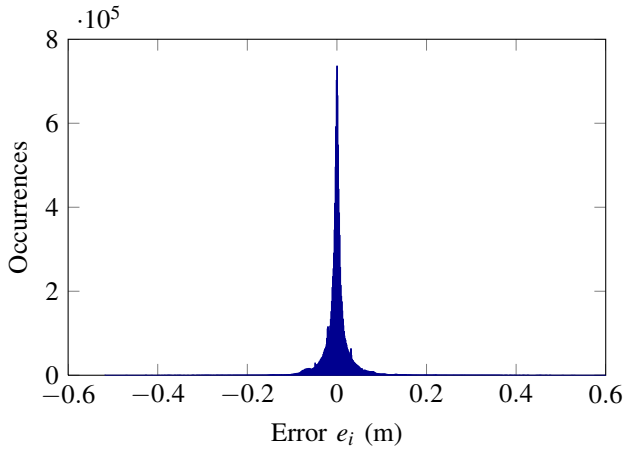

Fig. 4: Histogram of the error e_i for all the points p_i in our data set. This shows that a normal distribution is a good approximation to our data.

V. Conclusion

This work has presented a novel methodology for generating a measurement noise model for Kinect-style sensors. Classical work in sensor modeling has defined the uncertainty to be constant for all measurements. For many depth sensors, especially those like Kinect, this constant assumption simply isn't true. In recent work ad hoc models have been created to be more realistic. Our method generates sensor models from a thorough stochastic analysis of a large data set. We give our generated model and compare the results with a validation set. Our method is better than an ad hoc guess, however a model may need to be generated for a specific sensor depending on the needs of the algorithm it will be used for. We have made it easy to generate a model for your own sensor by making the code freely available. In future work we intend to improve our model by taking into account the error induced by lens distortion. Additionally, we would like to evaluate our model with existing SLAM algorithms.

Acknowledgment

The authors would like to thank professors Antônio W. Vieira and Renato Assunção for their valuable theoretical input. This work is partially supported by NSF grant OISE #1131305 and Sandia National Laboratories under PO 1179196, and by the brazilian institutes CAPES, CNPq and FAPEMIG.

References

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[2] J. Borenstein and L. Feng, "Correction of systematic odometry errors in mobile robots," in *IROS '95*, 1995, pp. 569–574.

[3] J. Wu and H. Zhang, "Camera sensor model for visual slam," in *Fourth Canadian Conference on Computer and Robot Vision, 2007. CRV '07.*, may 2007, pp. 149 –156.

[4] L. Clemente, A. Davison, I. Reid, J. Neira, and J. Tardós, "Mapping large loops with a single hand-held camera," in *Robotics: Science and Systems*, 2007.

[5] P. Newman, G. Sibley, M. Smith, M. Cummins, A. Harrison, C. Mei, I. Posner, R. Shade, D. Schroeter, L. Murphy, *et al.*, "Navigating, recognizing, and describing urban spaces with vision and lasers," *IJRR*, vol. 28, no. 11-12, pp. 1406–1433, 2009.

[6] S. May, D. Droeschel, D. Holz, S. Fuchs, E. Malis, A. Nüchter, and J. Hertzberg, "Three-dimensional mapping with time-of-flight cameras," *Journal of field robotics*, vol. 26, no. 11-12, pp. 934–965, 2009.

[7] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli, "Depth mapping using projected patterns," Patent Application WO 2008/120 217 A2, Oct. 9, 2008.

[8] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *IEEE International Symposium on Mixed and Augmented Reality*, vol. 7, no. 10, Imperial College London, UK. IEEE, 2011, pp. 127–136.

[9] L. Xia, C.-c. Chen, and J. K. Aggarwal, "Human Detection Using Depth Information by Kinect," in *International Workshop on Human Activity Understanding from 3D Data*, Colorado Springs, CO, 2011.

[10] J. Stowers, M. Hayes, and A. Bainbridge-Smith, "Altitude control of a quadrotor helicopter using depth map from Microsoft Kinect sensor," *2011 IEEE International Conference on Mechatronics*, pp. 358–362, Apr. 2011.

[11] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments," *IJRR*, 2012.

[12] M. F. Fallon, H. Johannsson, and J. J. Leonard, "Efficient scene simulation for robust monte carlo localization using an RGB-D camera," in *ICRA 2012*, 2012, pp. 1663–1670.

[13] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," *Autonomous robot vehicles*, vol. 1, pp. 167–193, 1990.

[14] M. Kaess, A. Ranganathan, and F. Dellaert, "isam: Incremental smoothing and mapping," *Robotics, IEEE Transactions on*, vol. 24, no. 6, pp. 1365–1378, 2008.

[15] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping (slam): Part i the essential algorithms," *Robotics and Automation Magazine*, vol. 13, no. 99, p. 80, 2006.

[16] T. Kahlmann, F. Remondino, and H. Ingensand, "Calibration for increased accuracy of the range imaging camera swissranger™," *Proc of IEVM*, vol. 36, no. 4, pp. 136–141, 2006.

[17] F. Chiabrando, R. Chiabrando, D. Piatti, and F. Rinaudo, "Sensors for 3D Imaging: Metric Evaluation and Calibration of a CCD/CMOS Time-of-Flight Camera." in *Sensors (Basel, Switzerland)*, vol. 9, no. 12, Jan. 2009, pp. 10 080–96.

[18] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer Magazine, Special Issue on Autonomous Intelligent Machines*, vol. 22, pp. 46–57, 1989.

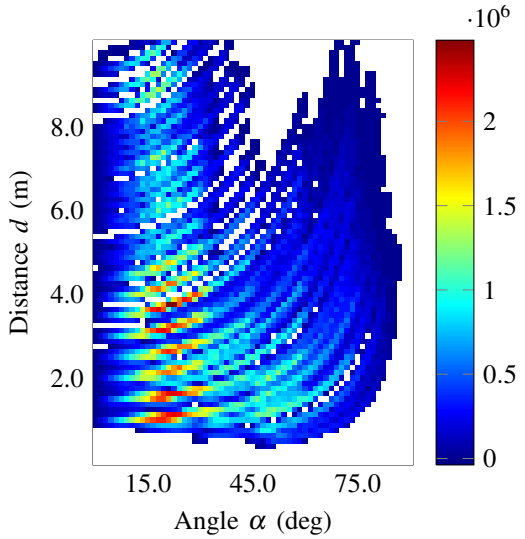

Fig. 5: Number of points in each neighborhood N_j. In this image, each neighborhood is represented by a pixel centered at the neighborhood centroid.

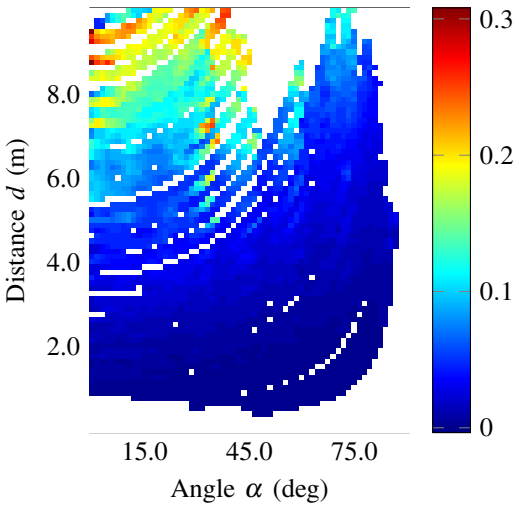

Fig. 7: Values for standard deviation σ_j at each neighborhood. We can see the variance increases with larger values distance d while being weakly influenced by the incident angle α.

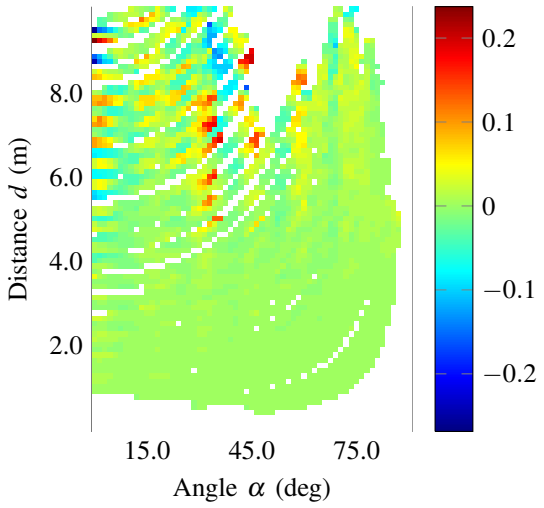

Fig. 6: Mean value of e_i in each neighborhood N_j which we define as $\overline{e_k}$. In our analysis we set this value equal to zero. This graph gives some experimental justification for this assumption.

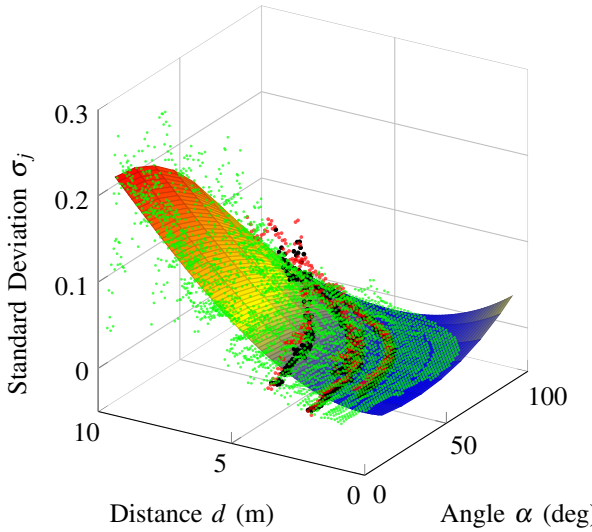

Fig. 8: Our noise model for the Kinect. The dots represent the local estimation of σ for a particular neighborhood N_j.