Ego-Motion Estimation on Range Images using High-Order Polynomial Expansion

Brian Okorn and Josh Harguess
Space and Naval Warfare Systems Center Pacific
San Diego, CA, USA
{brian.okorn,joshua.harguess}@navy.mil

Abstract

This paper presents two novel algorithms for estimating the (local and global) motion in a series of range images based on a polynomial expansion. The use of polynomial expansion has been quite successful in estimating optical flow in 2D imagery, but has not been used extensively in 3D or range imagery. In both methods, each range image is approximated by applying a high-order polynomial expansion to local neighborhoods within the range image. In the local motion algorithm, these approximations are then used to derive the translation or displacement estimation within the local neighborhoods from frame to frame within the series of range images (also known as range image flow). An iterative method for computing the local translations is presented. In the global motion algorithm, a global motion model framework is utilized to compute a global motion estimation based on the polynomial expansion of the range images. We evaluate the algorithms on several real-world range image sequences with promising results.

1. Introduction

Estimating motion in a video or series of images is an extremely important and difficult task in computer vision and has numerous applications, such as autonomous vehicle navigation [11]. Not much attention has been given to the estimation of motion between range images, however, estimating motion in video or image sequences, most commonly referred to as optical flow, has a long history of research. The two most common and prominent approaches to optical flow are known as local (or sparse) and global (or dense) methods. Local methods, such as the Lucas–Kanade method [14], estimate the motion of regions of interest between images using image registration and warping techniques. In contrast, global methods, such as Horn and Schunck's method [13], compute a dense motion field by estimating the motion of each pixel between images. In this work, we are concerned mostly with the latter method, particularly the work by Farnebäck [8], which approximates the image using a polynomial expansion of local patches and then uses the polynomial expansion to estimate the global displacements between images for each pixel.

Estimating optical flow for range images, also known as range flow, is the main topic of this paper. Very little current research exists on this topic, however, it is an extremely important problem in a growing field. One of the few examples is the work of Spies, Jahne and Barron [18]. The problem of range flow is unique from optical flow for electro-optical images in the sense that every pixel value is a measure of distance instead of color or brightness. This difference makes it very difficult to apply existing and traditional two-dimensional (2D) optical flow methods to range flow. For instance, the brightness constancy constraint used by many optical flow methods is not valid for range images. Therefore, our approach in this work is to extend a well-known global optical flow method of motion estimation based on polynomial expansion [8] to range images. We extend the method by using a high-order polynomial expansion to include terms in the z direction (range distance to the sensor). We then formulate an iterative method to solve for displacement in the x, y and z directions between range imagery. In addition, we perform the calculation at multiple scales for robustness and include displacement estimates from previous frames to improve the overall motion estimation. We also introduce a method to estimate the global motion based on motion model derivations simlar to that presented by Dufaux and Moscheni [4] and Farnebäck and Westin [10]. Promising results are presented on several real-world range images.

2. Related Work

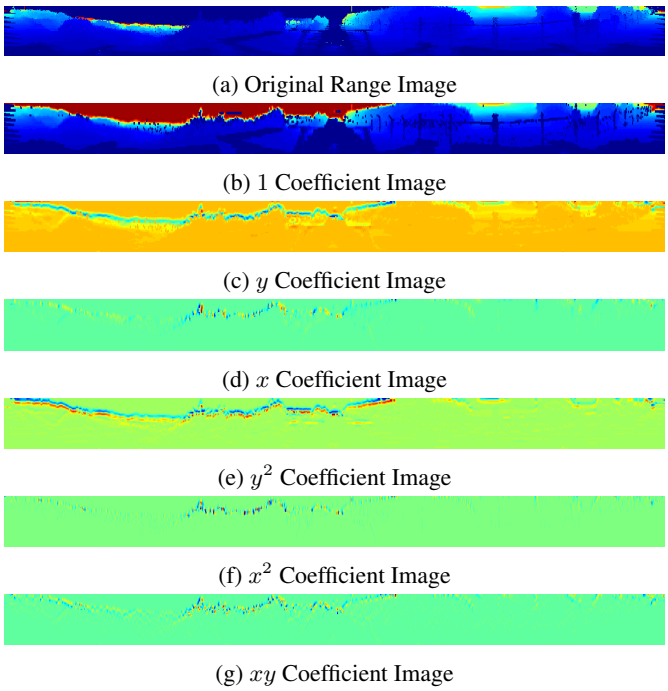
As previously mentioned, there is a vast amount of research in optical flow between color images, usually categorized into local and global methods. Many assumptions and constraints have been introduced in both approaches to

deal with noise and smoothness of the solutions, such as the brightness constancy assumption, gradient constancy assumption and spatio-temporal smoothness constraints. This has led to a breeding ground of methods, such as Bruhn et al. [3], which attempt to combine the local and global methods to address the drawbacks and assumptions of each individual method. The most popular and successful methods are covered in more detail in two benchmarking papers on the subject by McCane et al. [15] and Baker et al. [1]. The two papers describe common databases, procedures and results on comparing more than 20 optical flow methods, with the Baker et al. paper being the most recent and complete. Of particular interest to our research is the method introduced by Farnebäck in several papers that introduces the estimation of motion using the polynomial expansion [8, 9, 5, 7, 6, 19, 10, 17, 16]. In Farnebäck's work, the local neighborhoods of each image are approximated by a polynomial expansion and an analytical solution for the displacement between images is derived. From this derivation, a robust algorithm is designed to compute the displacement, and thus motion field, between two or more images in a sequence. The method has proven to be very accurate and robust for 2D images and has been included as a default algorithm in the OpenCV library [2].

The research of estimating the motion between range images, or range flow, is much more sparse. The term "range image flow" first appears in the work of Gonzalez [12], where he formulates a physics-based approach to estimate the motion of the range sensor relative to its environment. Our method uses the same basic physical model of the range sensor as that used by Gonzalez. One of the most popular and earliest papers on this topic, by Spies et al. [18], notes the unique challenges of this problem and proposes a basic motion constraint equation on deformable surfaces. The constraint solutions are obtained in a total least squares framework and compute a dense range flow field from sparse solutions. While the results are promising for the range images presented in their paper, the method is not directly transferable to other domains, such as dense range flow in a moving scene due to the large displacements present. Therefore, the focus of our work is to extend the polynomial expansion method to range imagery to compute local and global dense range flow on sequences of real-world range images.

3. Range Image Polynomial Expansion

In our formulation, range image flow uses a polynomial expansion based approximation of the range image. This approximation is done using a set of quadratic basis functions, applied to the range data. The basis equation set is $\{1, x, y, x^2, y^2, xy\}$, which describe the variation of z, range from the sensor, as you vary x and y, azimuth and elevation with respect to the sensor. In addition to the basis functions, we incorporate a notion of the accuracy or importance of this data though a certainty matrix, as well as a proximity-based weight over the neighborhood in the form of an applicability matrix, as is done in Farnebäck's Ph.D. dissertation [8]. For the certainty matrix, a value of 1 was given to all pixels populated with valid data and 0 for any pixel with error or no data. A Gaussian kernel was used as the applicability measure. The weights of the basis, $\{r_1, r_x, r_y, r_{x^2}, r_{y^2}, r_{xy}\}$, were calculated for Equation (1) described by Farnebäck [8] and the values of these weights for a Velodyne® lidar range image can be seen in Figure 1.



(a) Original Range Image

(b) 1 Coefficient Image

(c) y Coefficient Image

(d) x Coefficient Image

(e) y^2 Coefficient Image

(f) x^2 Coefficient Image

(g) xy Coefficient Image

Figure 1: Velodyne® polynomial expansion.

$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c \tag{1}$$

Using this formulation, the range, $f(\mathbf{x})$, is described as a function of pixel location, with $\mathbf{A} = \begin{bmatrix} r_{x^2} & \frac{r_{xy}}{2} \\ \frac{r_{xy}}{2} & r_{y^2} \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} r_x r_y \end{bmatrix}^T$ and $c = r_1$.

4. Local Flow Displacement Calculation

The polynomial expansion in 2D images allows displacement to be calculated analytically, by looking at the effects of a displacement on the polynomial expansion coefficients. The effects of this displacement on the quadratic polynomial are derived in Equation (2).

$$
\begin{aligned}
f_1(\mathbf{x}) &= \mathbf{x}^T\mathbf{A_1}\mathbf{x} + \mathbf{b_1}^T\mathbf{x} + c_1 \\
f_2(\mathbf{x}) &= f_1(\mathbf{x} - \mathbf{d}) \\
&= \mathbf{x}^T\mathbf{A_1}\mathbf{x} + (\mathbf{b_1} - 2\mathbf{A_1}\mathbf{d})^T\mathbf{x} + \mathbf{d}^T\mathbf{A_1}\mathbf{d} \\
&\quad + \mathbf{b_1}^T\mathbf{d} + c_1 \\
&= \mathbf{x}^T\mathbf{A_2}\mathbf{x} + \mathbf{b_2}^T\mathbf{x} + c_2,
\end{aligned} \tag{2}
$$

leaving you with a new quadratic polynomial with different coefficients,

$$\mathbf{A_2} = \mathbf{A_1} \tag{3}$$

$$\mathbf{b_2} = \mathbf{b_1} - 2\mathbf{A_1}\mathbf{d} \tag{4}$$

$$c_2 = \mathbf{d}^T\mathbf{A_1}\mathbf{d} + \mathbf{b_1}^T\mathbf{d} + c_1. \tag{5}$$

With these new coefficients, $\mathbf{d}$ can be computed from Equation (4).

$$\mathbf{d} = -\frac{1}{2}\mathbf{A_1}^{-1}(\mathbf{b_2} - \mathbf{b_1}) \tag{6}$$

This method of displacement calculation was developed and tested by Farnebäck and is used as the starting point of the three-dimensional (3D) displacement calculation.

A third dimension cannot simply be added to $\mathbf{d}$ because with the current model, it has no meaning as an input term. The function space is not defined for any value of z under this model so the model must be modified to explain behavior while displacing this third dimension.

To properly capture this higher-dimensional behavior, a higher-order equation is used to approximate this space. A linear spreading of the x and y data, Equation (7), as well as a constant increase, Equation (8), was used to approximate the behavior of the data as you move along the z dimension. This leads to a quartic polynomial, Equation (9), with the higher-order terms being very sparse tensors.

$$
\begin{aligned}
x' &= (\zeta_x z + 1)x \\
y' &= (\zeta_y z + 1)y \\
\zeta_x &= \alpha_x / z \\
\zeta_y &= \alpha_y / z \\
\alpha_x &= \frac{\text{angular range of } x}{\text{pixel range of } x} = \frac{2\pi \text{ radians}}{\text{image width}} \\
\alpha_y &= \frac{\text{angular range of } y}{\text{pixel range of } y} = \frac{26.8^\circ}{\text{image height}} \\
\mathbf{x'_2} &= \begin{bmatrix} x' \\ y' \end{bmatrix}
\end{aligned} \tag{7}
$$

$$f(\mathbf{x}) = f_{quad}(\mathbf{x'_2}) + z; \tag{8}$$

$$
\begin{aligned}
f(\mathbf{x}) &= \zeta_x^2 r_{x^2} x^2 z^2 + \zeta_x\zeta_y r_{xy} xyz^2 + \zeta_y^2 r_{y^2} y^2 z^2 \\
&\quad + 2\zeta_x r_{x^2} x^2 z + (\zeta_x + \zeta_y) r_{xy} xyz + 2\zeta_y r_{y^2} y^2 z \\
&\quad + r_{x^2} x^2 + r_{xy} xy + \zeta_x r_x xz + r_{y^2} y^2 + \zeta_y r_y yz \\
&\quad + r_x x + r_y y + z + c
\end{aligned} \tag{9}
$$

where ζ_x and ζ_y were derived from the planar projection model to account for the spreading of the points in the spherical coordinate system of the sensor.

This can be combined into a tensor form:

$$f(\mathbf{x}) = g_{ijkl}x_i x_j x_k x_l + h_{ijk}x_i x_j x_k + a_{ij}x_i x_j + b_i x_i + c, \tag{10}$$

where the high-order tensors, $\mathbf{G}$ and $\mathbf{H}$, are sparse and the quadratic tensors are dense. For $\mathbf{G}$ and $\mathbf{H}$, the nonzero terms are

$$
\begin{aligned}
g_{0022} &= g_{0202} = g_{2002} = \ldots = \frac{\zeta_x^2 r_{x^2}}{6} \\
g_{0122} &= g_{0212} = g_{2012} = \ldots = \frac{\zeta_x\zeta_y r_{xy}}{12} \\
g_{1122} &= g_{1212} = g_{2112} = \ldots = \frac{\zeta_y^2 r_{y^2}}{6}
\end{aligned} \tag{11}
$$

$$
\begin{aligned}
h_{002} &= h_{020} = \ldots = \frac{2\zeta_x r_{x^2}}{3} \\
h_{012} &= h_{120} = \ldots = \frac{(\zeta_x + \zeta_y) r_{xy}}{6} \\
h_{112} &= h_{121} = \ldots = \frac{2\zeta_y r_{y^2}}{3},
\end{aligned} \tag{12}
$$

while $\mathbf{A}$ and $\mathbf{b}$ are

$$\mathbf{A} = \begin{bmatrix} r_{x^2} & \frac{r_{xy}}{2} & \frac{\zeta_x r_x}{2} \\ \frac{r_{xy}}{2} & r_{y^2} & \frac{\zeta_y r_y}{2} \\ \frac{\zeta_x r_x}{2} & \frac{\zeta_y r_y}{2} & 0 \end{bmatrix} \tag{13}$$

$$\mathbf{b} = \begin{bmatrix} r_x & r_y & 1 \end{bmatrix}^T. \tag{14}$$

We now explore the effects of displacement as Farnebäck did in Equation (2):

$$
\begin{aligned}
f(\mathbf{x}) &= g_{ijkl}x_ix_jx_kx_l + h_{ijk}x_ix_jx_k + a_{ij}x_ix_j + b_ix_i + c \\
\tilde{f}(\mathbf{x}) &= f(\mathbf{x}-\mathbf{d}) \\
&= g_{ijkl}x_ix_jx_kx_l \\
&\quad - (4g_{ijkl}d_l - h_{ijk})x_ix_jx_k \\
&\quad + (6g_{ijkl} - 3h_{ijk}d_k + a_{ij})x_ix_j \\
&\quad - (4g_{ijkl}d_jd_kd_l - 3h_{ijk}d_jd_k + 2a_{ij}d_j - b_i)x_i \\
&\quad + (g_{ijkl}d_id_jd_kd_l - h_{ijk}d_id_jd_k + a_{ij}d_id_j - b_id_i + c \\
&= \tilde{g}_{ijkl}x_ix_jx_kx_l + \tilde{h}_{ijk}x_ix_jx_k + \tilde{a}_{ij}x_ix_j + \tilde{b}_ix_i + \tilde{c},
\end{aligned} \tag{15}
$$

leaving us with a new quartic polynomial with the following coefficients:

$$
\begin{aligned}
\tilde{g}_{ijkl} &= g_{ijkl} \\
\tilde{h}_{ijk} &= h_{ijk} - 4g_{ijkl}d_l \\
\tilde{a}_{ij} &= a_ij - 3h_{ijk}d_k + 6g_{ijkl}d_kd_l \\
\tilde{b}_i &= b_i - 2a_ijd_j + 3h_{ijk}d_jd_k - 4g_{ijkl}d_jd_kd_l \\
\tilde{c} &= c - b_id_i + a_ijd_id_j - h_{ijk}d_id_jd_k + g_{ijkl}d_id_jd_kd_l.
\end{aligned} \tag{16}
$$

Now we have a linear equation with $\mathbf{H}$ that could be used as an analytical solution to $\mathbf{d}$ but with our higher-order tensors being mostly sparse and mainly composed of the model coefficients as opposed to the expansion coefficients, we look to our lower order dense tensors to solve for $\mathbf{d}$ through numerical optimization. Using the symmetries and sparsities of the matrices, we are left with the following coefficients in our dense tensors.

$$
\begin{aligned}
\tilde{b}_0 &= b_0 - 2(a_{00}d_x + a_{01}d_y + a_{02}d_z) \\
&\quad + 6(h_{002}d_xd_z + h_{012}d_yd_z) \\
&\quad - 12(g_{0022}d_xd_z^2 + g_{0122}d_yd_z^2) \\
\tilde{b}_1 &= b_1 - 2(a_{01}d_x + a_{11}d_y + a_{12}d_z) \\
&\quad + 6(h_{012}d_xd_z + h_{112}d_yd_z)) \\
&\quad - 12(g_{0122}d_xd_z^2 + g_{1122}d_yd_z^2) \\
\tilde{b}_2 &= b_2 - 2(a_{02}d_x + a_{12}d_y) \\
&\quad + 3(h_{002}d_x^2 + h_{112}d_y^2 + 2h_{012}d_xd_y) \\
&\quad - 12(g_{0022}d_x^2d_z + g_{1122}d_y^2d_z + 2g_{0122}d_xd_yd_z)
\end{aligned} \tag{17}
$$

$$
\begin{aligned}
\tilde{c} &= c - b_0d_x + b_1d_y + b_2d_z \\
&\quad + (a_{00}d_x^2 + a_{11}d_y^2 + 2a_{01}d_xd_y + 2a_{02}d_xd_z + 2a_{12}d_yd_z) \\
&\quad - 3(h_{002}d_x^2d_z + h_{112}d_y^2d_z + 2h_{012}d_xd_yd_z) \\
&\quad + 6(g_{0022}d_x^2d_z^2 + g_{1122}d_y^2d_z^2 + 2g_{0122}d_xd_yd_z^2).
\end{aligned} \tag{18}
$$

To calculate $\mathbf{d}$, we optimize the difference between the observed polynomial coefficients of the next image and the coefficients derived through displacing the coefficients of the first image using Equations (17) and (18). This optimization is done using the non-linear least squares version of Newton's method, the Gauss-Newton algorithm, on Equations (19) and (20). This optimization technique requires the Jacobian matrices of each coefficient with respect to $\mathbf{d}$.

$$
\min_{\mathbf{d}} \sum (\Delta b_i)^2, \qquad \Delta\mathbf{b} = \tilde{\mathbf{b}}^{(1)}(\mathbf{d}) - \mathbf{b}^{(2)} \tag{19}
$$

$$
\min_{\mathbf{d}} \sum (\Delta c)^2, \qquad \Delta c = \tilde{b}^{(1)}(\mathbf{d}) - c^{(2)} \tag{20}
$$

This leaves two remaining routes to solving for $\mathbf{d}$, through the changes in $\mathbf{b}$ or the changes in c, shown in Equations (21) and (22), respectively, each with its own advantage. The solution based on $\mathbf{b}$ tends to produce more accurate results for d_x and d_y because it captures the motion of the quadratic components seen mostly in edges. This solution does not tend to be as accurate for the d_z component because d_z is detected through the spreading of the quadratic x and y components. This effect can be small at long distances, where as d_z has a large effect on c. However, the effects on c do not track the quadratic components of d_x and d_y as directly as $\mathbf{b}$.

$$
\Delta\mathbf{d}_\mathbf{b} = (\mathbf{J}_{\mathbf{b}(\mathbf{x})}{}^T\mathbf{J}_{\mathbf{b}(\mathbf{x})})^{-1}\mathbf{J}_{\mathbf{b}(\mathbf{x})}{}^T\Delta\mathbf{b} \tag{21}
$$

$$
\Delta\mathbf{d}_\mathbf{c} = (\mathbf{J}_{\mathbf{c}(\mathbf{x})}{}^T\mathbf{J}_{\mathbf{c}(\mathbf{x})})^{-1}\mathbf{J}_{\mathbf{c}(\mathbf{x})}{}^T\Delta c \tag{22}
$$

Similar to Farnebäck, this algorithm estimates the displacement over a neighborhood I around $\mathbf{x}$ as opposed to a single point, minimizing the following equations:

$$
\sum_{x\in I} w_\mathbf{x}\|(\tilde{b}_i^{(1)}(\mathbf{x}+\Delta\mathbf{x}-\mathbf{d}) - b_i^{(2)}(\mathbf{x}+\Delta\mathbf{x}))\|^2 \tag{23}
$$

$$
\sum_{x\in I} w_\mathbf{x}\|(\tilde{c}^{(1)}(\mathbf{x}+\Delta\mathbf{x}-\mathbf{d}) - c^{(2)}(\mathbf{x}+\Delta\mathbf{x}))\|^2, \tag{24}
$$

where $w_\mathbf{x}$ is the neighborhood weighting function and the minimum steps are

$$
\Delta\mathbf{d}_\mathbf{b} = \left(\sum_{\mathbf{x}\in I} w_\mathbf{x}\mathbf{J}_{\mathbf{b}(\mathbf{x})}{}^T\mathbf{J}_{\mathbf{b}(\mathbf{x})}\right)^{-1} \sum_{\Delta x\in I} w_\mathbf{x}\mathbf{J}_{\mathbf{b}(\mathbf{x})}{}^T\Delta\mathbf{b} \tag{25}
$$

$$
\Delta\mathbf{d}_\mathbf{c} = \left(\sum_{\mathbf{x}\in I} w_\mathbf{x}\mathbf{J}_{\mathbf{c}(\mathbf{x})}{}^T\mathbf{J}_{\mathbf{c}(\mathbf{x})}\right)^{-1} \sum_{\Delta x\in I} w_\mathbf{x}\mathbf{J}_{\mathbf{c}(\mathbf{x})}{}^T\Delta c. \tag{26}
$$

5. Global Flow Displacement Calculation

To compute the global motion estimation, we utilize the material covered in Sections 3 and 4 to compute the polynomial expansion of the range images which are utilized in a global motion model. We follow the derivation and notation of Farnebäck and Westin [10] and derive the global motion estimation in the following section.

We model the displacement as $\mathbf{d}(\mathbf{x}) = \mathbf{S}(\mathbf{x})\mathbf{p}$, where $\mathbf{d}(\mathbf{x})$ is the displacement estimation, $\mathbf{S}(\mathbf{x})$ is the motion model, and $\mathbf{p}$ are the translation and rotation parameters, $\mathbf{p} = \begin{bmatrix} T_x & T_y & T_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^T$, which are optimized to produce the global motion estimation. Using the work of Dufaux and Moscheni [4] and the spherical coordinate system inherent to our sensor, we construct a motion model $\mathbf{S}(\mathbf{x})$ for our Velodyne® sensor.

The final motion model for our sensor is

$$\mathbf{S}(\mathbf{x}) = \begin{bmatrix} \mathbf{S}_{size}\mathbf{S}_T & \mathbf{S}_{size}\mathbf{S}_R \end{bmatrix} \tag{27}$$

where

$$\mathbf{S}_{size} = \begin{bmatrix} \frac{1}{\alpha_x} & 0 & 0 \\ 0 & \frac{1}{\alpha_y} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{28}$$

$$\mathbf{S}_T = \begin{bmatrix} \frac{sin(\theta)}{rsin(\phi)} & \frac{-cos(\theta}{rsin(\phi)} & 0 \\ \frac{-cos(\theta)cos(\phi)}{r} & \frac{-sin(\theta)cos(\phi)}{r} & \frac{-sin(\phi)}{r} \\ -cos(\theta)sin(\phi) & -sin(\theta)sin(\phi) & cos(\phi) \end{bmatrix}, \tag{29}$$

$$\mathbf{S}_R = \begin{bmatrix} \frac{-cos(\theta)cos(\theta)}{sin(\theta)} & \frac{-sin(\theta)cos(\phi)}{sin(\phi)} & -1 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{30}$$

The matrix $\mathbf{S}_{size}$ accounts for the scaling of each pixel where α_x and α_y are defined as in Equation 7. The matrices $\mathbf{S}_T$ and $\mathbf{S}_R$ account for the translational and rotational components for each pixel $[x_i, y_i]$ of the motion model, respectively, where

$$\begin{aligned} \theta &= \alpha_x x_i + \text{minimum azimuth} \\ \phi &= \alpha_y y_i + \left(\text{minimum elevation} + \frac{\pi}{2}\right). \end{aligned} \tag{31}$$

We set up a (nonlinear) least squares problem using the steps outlined in [10] where the Gauss-Newton solution for p is found to be (iterating over $p = p - \Delta p$)

$$\begin{aligned} \Delta p = &\left(\sum \beta_1 \mathbf{S}^T \mathbf{J}_c \mathbf{J}_c{}^T + \mathbf{S} + \beta_2 \mathbf{S}^T \mathbf{J}_b \mathbf{J}_b{}^T \mathbf{S}\right)^{-1} \times \\ &\left(\sum \beta_1 \mathbf{S}^T \mathbf{J}_c{}^T \Delta c + \beta_2 \mathbf{S}^T \mathbf{J}_b{}^T \Delta \mathbf{b}\right) \end{aligned} \tag{32}$$

where $\mathbf{J}_c$ and $\mathbf{J}_b$ are the Jacobians of c (linear components) and b (non-linear components) with respect to d, respectively. The variables Δc and Δb are the residuals from $(c_1 - \tilde{c_2})$ and $(b_1 - \tilde{b_2})$, respectively.

6. Range Image Flow Experiments

The local and global algorithms have been tested using data from a Velodyne® HDL-64E. This sensor is a 360° field of view 3D lidar with 64 vertically mounted lasers on a spinning head. The lasers have a maximum range of 50 m and an accuracy of 2 cm. It is capable of spinning at 5 to 15 Hz, generating over 1.333 million points per second. For our tests, the sensor was set to 10 Hz, generating a horizontal resolution of 1800 returns per rotation. The lidar returns are assembled into a 64 x 1800 range image. The tests were done using a Ford F-150 with the Velodyne® mounted to the roof. The vehicle and mounting hardware are visible in the lidar scans, so all pixels within a threshold have been marked with a certainty value of 0, causing these values to have no effect on the polynomial expansion or the flow calculations.

While we don't yet have ground truth for this data, we did compute the Normal Distributions Transform (NDT) for the original point clouds to use for comparison. The data set shown in Figures 2a, 3a, 4a and 5a was collected around our facilities on a foggy day. The images contain mostly data from the bushes and other vegetation surrounding the road, but do eventually show a parking lot, seen in Figure 5a. The NDT results are overlaid on these images for use as comparison to our presented algorithms.

6.1. Local Motion Estimation

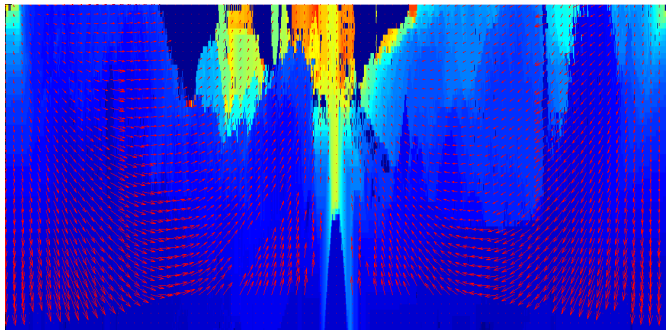
The results of the local motion estimation algorithm are shown in Figures 2b, 3b, 4b and 5b. Most of the flow field appears accurate, though some regions still contain peculiar behavior. Certain regions of the flow image act as sources or sinks to the flow fields, such as in Figure 3b, where in the middle right portion of the image the flow field moves away from a central point. The boundary of the scan also seems to be less accurate than the central areas. Areas such as that shown in Figure 4b are relatively uniform, making flow calculations more difficult. Despite this, the algorithm performed reasonably well, though it does have regions with sporadic flow behavior. Perhaps by combining the results of the global motion estimation with the local motion estimation, areas within the image that exhibit unique motion with respect to the platform may be identified as obstacles to aide in navigation.

6.2. Global Motion Estimation

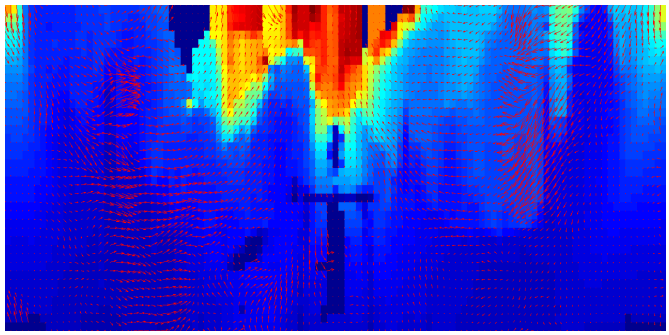
To test the efficacy of the global motion estimation algorithm, we first test on a known z rotation, since we can

create an exact rotation with the data synthetically by simply shifting the pixels in the range image to the left or right. The global range flow from this shift is shown in Figure 6 and the value of the translation and rotation parameters per iteration is shown in Figure 7. As seen in both figures, the correct solution is found and all of the motion is estimated to be in the z rotation.

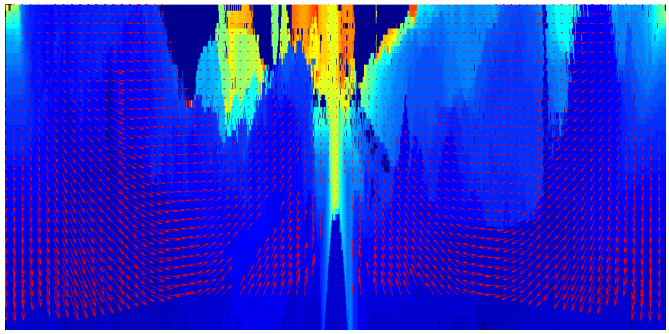
The results of the global motion estimation algorithm on the real-world data are shown in Figures 2c, 3c, 4c and 5c. As shown in the figures, the global motion estimation gives a consistent picture of the ego-motion of the platform between two successive range images. For instance, in Figure 2c, the flow is in the forward direction only, which corresponds to the correct action of the platform of a translation perpendicular to the image plane, as is confirmed by the results from the NDT algorithm in Figure 2a. However, in Figure 4, the results of the global motion estimation show little or no motion, which does not match up with the motion estimate of the NDT algorithm. Upon further inspection of the range images around frame 400, we found a lack of texture in the images which could explain this result. Also, we found it difficult even for a human to discern the motion present in those sequences of frames. Therefore, further testing on low texture range images is needed. While the global motion estimations may give more useful information for finding the ego-motion of the platform, estimating the local motion remains an important task for true autonomous navigation.



(a) NDT Flow Estimation

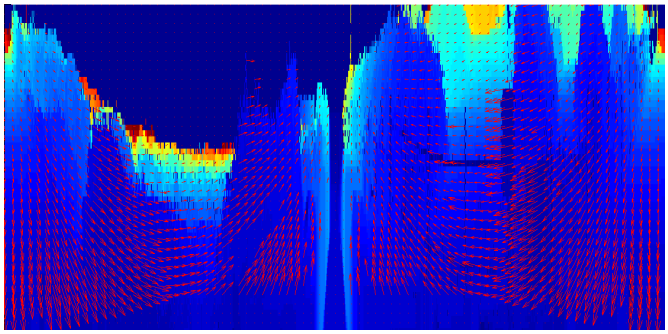


(b) Local Motion Estimation

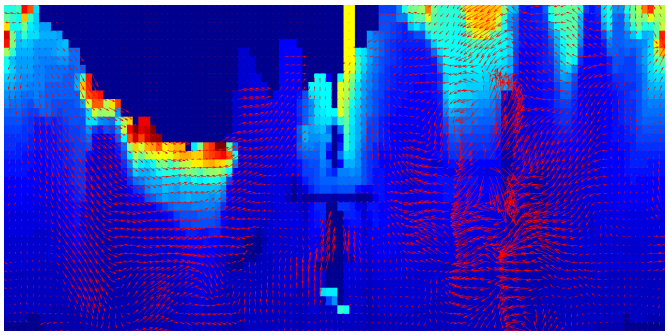


(c) Global Motion Estimation

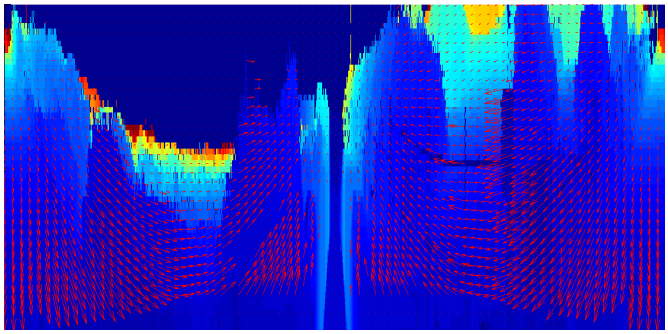
Figure 2: Range image flow on Velodyne® scan 100.



(a) NDT Flow Estimation

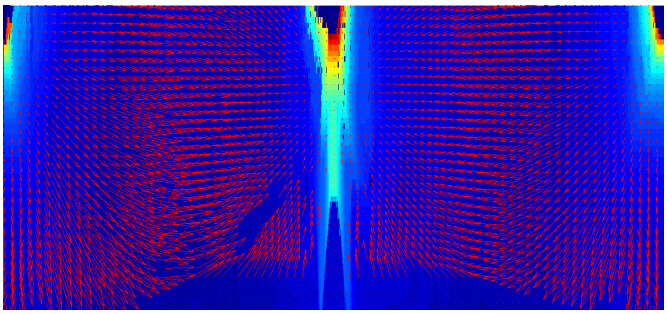


(b) Local Motion Estimation

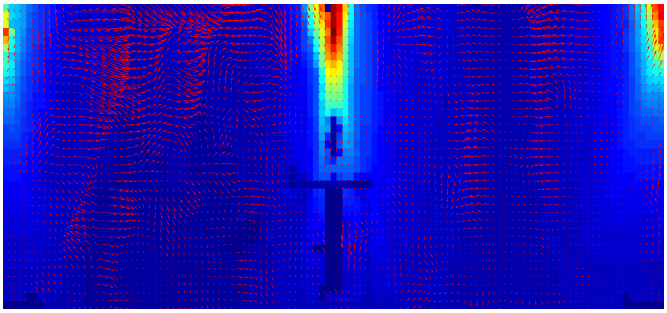


(c) Global Motion Estimation

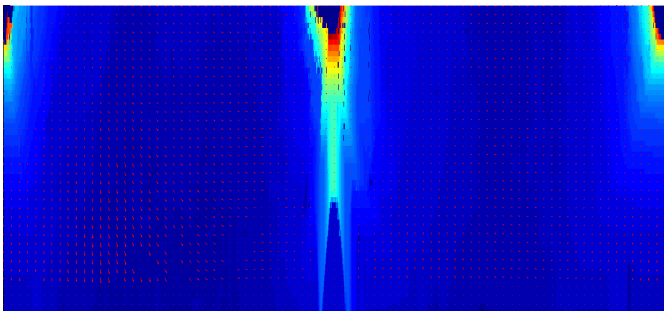
Figure 3: Range image flow on Velodyne® scan 204.



(a) NDT Flow Estimation

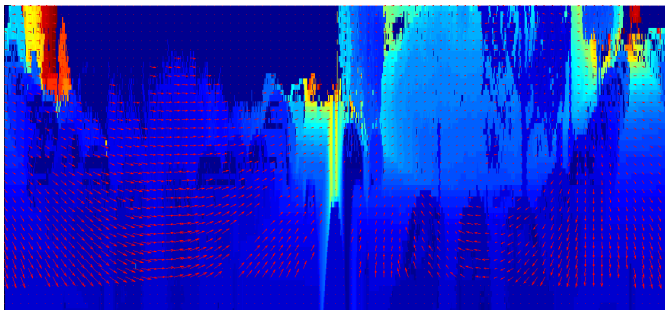


(b) Local Motion Estimation

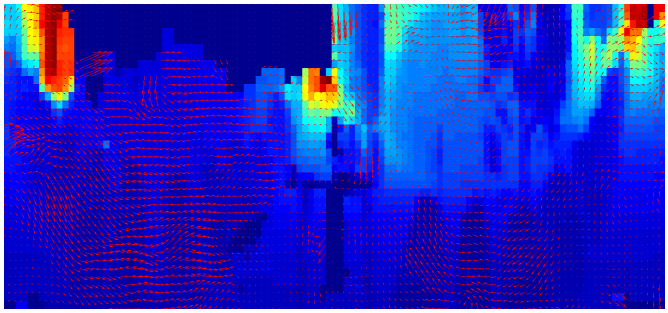


(c) Global Motion Estimation

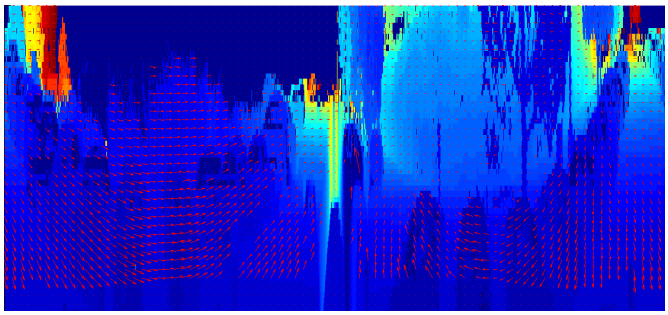
Figure 4: Range image flow on Velodyne® scan 400.



(a) NDT Flow Estimation



(b) Local Motion Estimation



(c) Global Motion Estimation

Figure 5: Range image flow on Velodyne® scan 1000.

7. Conclusion

The Polynomial Expansion-based method is effective in calculating the flow of range data, as our results have shown. The local motion estimation algorithm runs into some issues where regions of the range image appear as sources or sinks of motion, however, the overall results agree with the correct motion. The global motion estimation algorithm does not have these issues and appears to estimate the global motion accurately. For navigation applications, both local and global motion estimations will be useful to both estimate the ego-motion of the platform as well as estimate the movements of objects around the platform for avoidance and tracking. There may also be some improvements to overall motion estimation by combining the local and global estimations in a meaningful way. In future work, quantitative testing will be done to validate the qualitative results, using ground truth data to run large scale tests. Once complete, we are hoping to release this dataset to the wider research community. Additionally, the polynomial expansion may be used to segment the range image and classify planar and non-planar regions, which may then be incorporated into the flow calculation. Certain features have more stable information about their motion and as such should take on greater weights in the flow calculations. Planer regions, for example, contain very little information about the data's motion, but the edge and corner features contribute a large amount of flow information. Using the planar segmentation, the flows calculated at the corners and edges of planar regions could be interpolated across the region. In addition to the range image work, future work will focus on calculating the rotational components of the flow vectors with respect to a single image location, useful in image stabilization and unmanned aerial vehicle orientation determination.

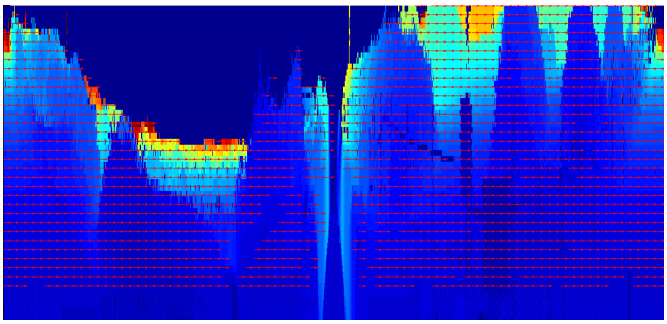


Figure 6: Global motion estimation of z rotation

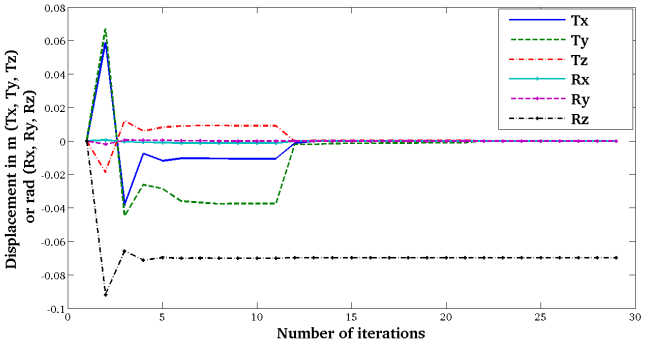


Figure 7: Plot of z rotation estimation per iteration

References

[1] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011. 2

[2] G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'reilly, 2008. 2

[3] A. Bruhn, J. Weickert, and C. Schnrr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):211–231, 2005. 2

[4] F. Dufaux and F. Moscheni. Segmentation-based motion estimation for second generation video coding techniques. In *Video Coding*, pages 219–263. Springer, 1996. 1, 5

[5] G. Farnebäck. Fast and accurate motion estimation using orientation tensors and parametric motion models. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 1, pages 135–139. IEEE, 2000. 2

[6] G. Farnebäck. Orientation estimation based on weighted projection onto quadratic polynomials. In *VMV*, pages 89–96, 2000. 2

[7] G. Farnebäck. Very high accuracy velocity estimation using orientation tensors, parametric motion, and simultaneous segmentation of the motion field. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 171–177. IEEE, 2001. 2

[8] G. Farnebäck. *Polynomial expansion for orientation and motion estimation*. PhD thesis, Linköping, 2002. 1, 2

[9] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Image Analysis*, pages 363–370. Springer, 2003. 2

[10] G. Farnebäck and C.-F. Westin. Affine and deformable registration based on polynomial expansion. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2006*, pages 857–864. Springer, 2006. 1, 2, 5

[11] A. Giachetti, M. Campani, and V. Torre. The use of optical flow for road navigation. *Robotics and Automation, IEEE Transactions on*, 14(1):34–48, 1998. 1

[12] J. Gonzalez. Recovering motion parameters from a 2d range image sequence. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 1, pages 433–440 vol.1, 1996. 2

[13] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial intelligence*, 17(1):185–203, 1981. 1

[14] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981. 1

[15] B. McCane, K. Novins, D. Crannitch, and B. Galvin. On benchmarking optical flow. *Computer Vision and Image Understanding*, 84(1):126–143, 2001. 2

[16] K. Nordberg and G. Farnebäck. A framework for estimation of orientation and velocity. In *Image Processing, 2003 International Conference on*, volume 3, pages III–57. IEEE, 2003. 2

[17] K. Nordberg and G. Farnebäck. Estimation of orientation tensors for simple signals by means of second-order filters. *Signal Processing: Image Communication*, 20(6):582–594, 2005. 2

[18] H. Spies, B. Jähne, and J. L. Barron. Range flow estimation. *Computer Vision and Image Understanding*, 85(3):209–231, 2002. 1, 2

[19] Y.-j. Wang, G. Farnebäck, and C.-F. Westin. Multi-affine registration using local polynomial expansion. *Journal of Zhejiang University SCIENCE C*, 11(7):495–503, 2010. 2