2017 IEEE Intelligent Vehicles Symposium (IV)
June 11-14, 2017, Redondo Beach, CA, USA

Monocular Localization within Sparse Voxel Maps

David Wong[1], Yasutomo Kawanishi[1], Daisuke Deguchi[2], Ichiro Ide[1], Hiroshi Murase[1]

Abstract— **We introduce a method that uses a single camera to localize a vehicle within a pre-constructed map consisting of a voxel occupancy grid and road-line marker positions. Sophisticated mapping hardware is capable of creating high-accuracy 3D maps of road environments, but localizing a vehicle within such maps is one of the challenges at the forefront of automated driving. A solution which is robust to dynamic environments, while using only inexpensive sensors, is a difficult problem. In addition, maps that enable precise localization consume a lot of data which is impractical for the expansive environments encountered in real-world road networks. We show how using the area of edge regions shared between rendered views of a compact voxel map and in-vehicle camera images can be coupled with non-linear optimization methods to determine the camera position and pose.**

I. Introduction

Vehicle self-localization is an increasingly important area of research, as we move towards fully autonomous driving systems. There are many approaches that aim to overcome the problem of ego-localization within a given map, each with their own sensor requirements and mapping techniques. For automotive localization, Global Positioning System (GPS) receivers are commonplace in navigation products. Aside from very expensive Real-Time Kinematic (RTK) systems, consumer GPS are not particularly accurate and can suffer from large errors when satellite signals are blocked or reflected off buildings and road infrastructure.

For more reliable localization, there are many combinations of sensors and mapping methodologies that can be applied. Important considerations for automotive localization systems include the cost and simplicity of the sensors used in localization, and the size and complexity of the required map. A single forward-facing camera provides an obvious choice as a readily available sensor, but creating a compact map which allows reliable visual localization is a challenge. Traditional methods include matching features extracted from images to create a feature-based map [1], [2], and image databases that use direct visual similarity to determine the camera's location [3], [4]. The increasing availability of inexpensive parallel processing power in GPUs has led to a number of direct localization techniques that render a dense 3D map for direct appearance comparison with input camera images [5]–[7]. In the field of medical imaging, a similar concept is used for the pose estimation of flexible endoscopes in surgical navigation [8], where renders of 3D scans are used for registration of endoscope images. Direct localization methods generally require a high level of detail in the rendering process, with dense 3D models leading to large database sizes.

Our approach employs direct localization techniques, but we base our solution around a compact map which contains much less information than most of the state-of-the-art methods. We use a map created by HERE [9] which contains a series of points representing corners of voxels that make up a coarse occupancy grid. The map provides no color or texture information, so common techniques that use joint image entropies can not be easily applied to the image registration problem. Instead, we apply an image gradient-based objective function that is minimized where mutual edges exist between rendered and real camera images. We then use a non-linear Levenberg-Marquardt [10] optimization to determine the camera pose within the map. The proposed system does not use lighting in the map rendering, and only mutual edges are included in the cost function, allowing camera localization which is robust to the lighting changes and occlusions typically found in traffic environments. Our main contributions can be summarized as follows:

1) We use a compact, texture-less sparse voxel map for rendering map scenes in order to perform localization with a monocular camera.
2) We employ a novel objective function that measures scene similarity based on the area of overlapping edge regions, formulated in a way that enables image comparison across a change in image modality. We also show how the partial derivatives of this function can be derived, allowing alignment of rendered voxel map images to real camera images within a least squares optimization framework.

This paper is organized as follows. In Section II we discuss research that is related to our approach. In Section III we briefly summarize the data and present the problem formulation. We outline our proposed method in Section IV and present the corresponding results in Section V. Following a discussion of the methods and results in Section VI, we conclude the paper in Section VII.

II. Related Work

Vision-based automotive localization research is often approached in a similar way to Simultaneous Localization and Mapping (SLAM) [11]. Many popular SLAM methods use repeatable feature points and descriptors, usually based on the original Scale Invariant Feature Transform (SIFT) [12]. Gradient histograms of local patches at extracted feature points are used to create descriptors that can be repeatably

[1]David Wong, Yasutomo Kawanishi, Ichiro Ide, and Hiroshi Murase are with the Graduate School of Information Science, Nagoya University, Japan davidw@murase.m.is.nagoya-u.ac.jp, {kawanishi, ide, murase}@is.nagoya-u.ac.jp

[2]Daisuke Deguchi is with the Information Strategy Office, Nagoya University, Japan ddeguchi@nagoya-u.jp

matched from different viewpoints, even with some changes in illumination. In SLAM, dense collections of feature points have been effective for small environments [13], and conceptually similar methods have been extended to the more expansive maps of automotive localization [1]. Feature-based methods may require many features for accurate pose estimation, since they use geometric constraints between matched features to calculate pose. Therefore, these databases can become very large. Even the best feature extractors and descriptors have limitations in terms of robustness to affine transformations and illumination changes.

Other visual localization techniques do not explicitly perform pose estimation and instead find the most similar image to the current query image within a database of pre-captured images. The image similarity metric can be determined using the Euclidean distance of dimension reduced images [14], or a low bit-rate image sequence instead of single images [4]. Features can also be used in these methods, by comparing matched feature descriptor similarity [15], or feature scale changes [2]. Image similarity methods are suitable for the predominantly linear motion of the automotive localization problem, but accuracy is limited by database density. Image databases with small image or image descriptor spacing become very large.

A natural extension of image matching methods is to generate database images using a 3D prior map, allowing any potential view within the map to be rendered. Such maps can be created using LiDAR scanners and calibrated cameras, giving textured photo realistic maps [7], or LiDAR reflectance data can provide photometric information instead of a camera [5], [6]. Localization is performed by employing optimization of the camera pose with an objective function relative to the current query image. Textured maps can employ a simple per-pixel Sum of Squared Distance (SSD) objective function [16], but where a modality change occurs (i.e. with LiDAR reflectance maps), mutual information between the virtual rendered camera and real camera images is commonly used as an objective function [17]. Mutual information is a measure of entropy correlation between the images. Normalized Information Distance (NID) is a true metric version of mutual information which has also been applied in pose estimation from rendered map views [5]–[7]. However, standard mutual information techniques are not simple to apply to the problem we present in this paper, where renderings of a texture-less voxel map contain only a fraction of the information of a camera image.

III. Problem Definition

In this paper we direct our attention to a specific map for automotive localization. Whereas other map rendering techniques typically use a dense textured map, the map that we employ here includes limited information in the form of painted road marking positions and a voxel occupancy grid.

A. The HERE Voxel Map Dataset

The voxel map adopted by the proposed method was initially distributed as part of the University Grand Challenge competition, which featured in the Intelligent Transport Systems World Congress 2016 (ITSWC2016) [18]. The dataset was supplied by HERE [9], and was captured on a multi-lane highway. The map consists of a set of cuboid voxel GPS coordinates, which define areas of the map close to the road that contain part of a solid object. While the road surface itself is not included in the occupancy grid, the location of painted road lines are provided as a string of GPS coordinates. Each voxel is approximately 25 cm square. While no information on how the map was created was provided, it was most likely constructed from LiDAR point clouds sub-sampled into voxel cubes, with road-lane marker positions extracted from reflectance information.

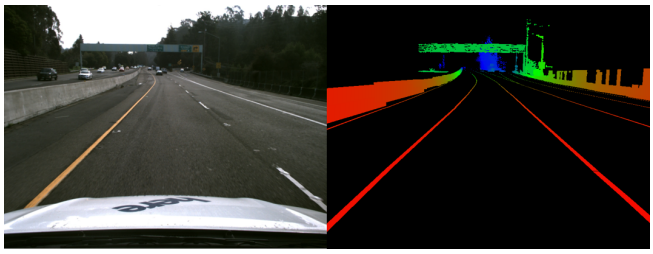

Fig. 1. An example of scene rendered from the voxel map, with its corresponding camera image on the left.

The dataset provides a series of timestamped query images captured from a forward-facing vehicle-mounted camera. While camera intrinsic calibration parameters are provided, no extrinsic information about the camera itself is available. Ground-truth information is provided as a set of camera capture locations. The camera images were captured at 10 Hz. Readings from a conventional 1 Hz GPS receiver are also supplied, but the accuracy is very low —with errors exceeding 30 m in places.

Fig. 1 shows an example rendered map scene, together with its corresponding query image.

B. Pose Optimization Formulation

The goal of this research is to determine the capture position of the query image relative to the voxel map. The extrinsic parameters of a virtual camera viewing the voxel map scene will align with the location and pose of the real camera when the rendered and real scenes share the most overlapping information. Determining these paramters is an $\mathbb{SE}^3$ optimization problem, and changes in Euler angles are directly correlated to any objective function used so they must be included in the optimization process.

The virtual camera pose $\mathbf{r} = [t_X, t_Y, t_Z, \theta_X, \theta_Y, \theta_Z]$ that is equal to the pose of the real camera $\bar{\mathbf{r}}$, can be determined by rendering a view of the map from an initial pose, comparing the rendered image $I(\mathbf{r}, \mathbf{x})$ to the real image $\bar{I}$ through a similarity metric, and then performing a gradient descent or Gauss-Newton optimization to find the parameters that make up the Euler angle representation of the camera pose that minimize the similarity metric function. The pose estimation becomes:

$$\hat{\mathbf{r}} = \arg\min_{\mathbf{r}} \rho(\bar{I}, I(\mathbf{r})), \tag{1}$$

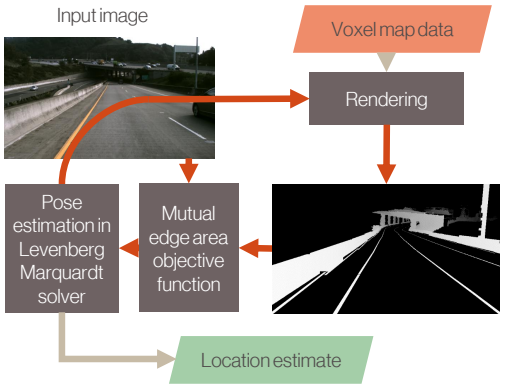

Fig. 2. An overview of the process of the proposed method.

where $\rho(\bar{I}, I(\mathbf{r}))$ defines the similarity metric. Typical similarity metrics, such as pixel-wise SSD and NID, are minimized when the real and rendered images are aligned.

The voxel map used in this paper has no texture information, so the voxels are rendered as a depth map, with the color of each vertex encoding the distance to the camera. The lack of texture and sparse nature of the rendered voxel map, as well as the change in modality between the real and rendered images, make this optimization problem poorly suited to metrics such as SSD and NID.

IV. Proposed Method

The proposed method performs a least squares optimization to determine camera pose, employing an objective function which measures the edge overlap of rendered and real images. An overview of the process for localizing an input image is shown in Fig. 2. The method for determining image similarity from mutual edge area is presented in Fig. 3.

In the subsequent sections, we describe the main process of the proposed method, which can be summarized as A) objective function formulation, B) optimization framework, and C) localization procedure.

A. Objective Function Formulation

The objective function must provide robustness to noise and a wide convergence basin. It can be observed from visual comparison of the real and rendered images that the majority of shared information lies in the mutual edge areas. However, where most objective functions use some kind of pixel-wise subtraction, the predominantly empty nature of the voxel map renders will provide a poor similarity metric. By combining these two observations, we propose a per-pixel multiplication of rendered and real camera images, as shown in Fig. 3.

We employ the Canny [19] operator to generate edge images for both the rendered and real camera images. The resulting objective function is as follows:

$$G(\mathbf{r}) = \left(\sum_{\mathbf{x} \in \bar{I}, I(\mathbf{r})} Q(\bar{I}) \circ Q(I(\mathbf{r})) \right)^{-1}, \qquad (2)$$

where $\mathbf{x} = [u, v]^{\mathrm{T}}$ denotes pixel location, and $Q(\bullet)$ is the Canny operator followed by a Gaussian blur over the entire

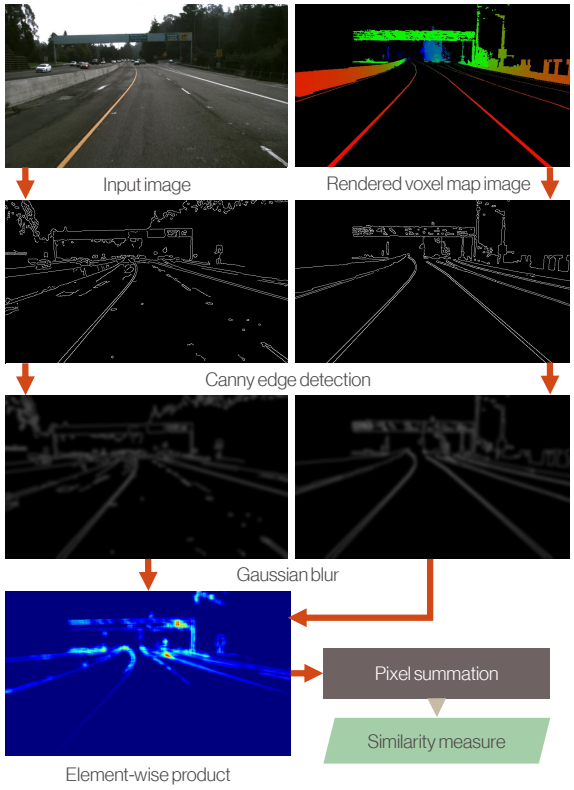

Fig. 3. The process for determining image similarity between camera images and voxel map renders, which is used by the mutual edge area objective function of the proposed method.

image. Note that we have to use the inverse of the sum rather than the inverse of individual elements to prevent division by zero. Strictly speaking, we could maximize $G(\mathbf{r})$ instead of minimizing $G(\mathbf{r})^{-1}$, but for implementation using optimization libraries, and for comparison with other minimization cost functions, we use the inverse function. The resulting objective function produces a clear minimum across the pose parameters, as shown in Fig. 4 (a) and (b).

B. Optimization Framework

The search space for the optimal parametrization that minimizes Eq. (2) covers six degrees of freedom, making it a problem that requires solving with an optimization method. While the proposed objective function could be used within any optimizer, we use the Ceres solver [20] implementation of the Levenberg-Marquardt [10] non-linear solver. Optimization methods usually use either gradient descent or Gauss-Newton to determine the parameters for the next iteration, which require the calculation of the Jacobian matrix. While the Ceres solver provides numeric differentiation, this is error prone and computationally intensive, so analytic derivation of the Jacobian matrix is desirable. The derivative of Eq. (2) requires differentiation of the similarity metric, which we proceed with using the product rule as follows:

$$\frac{\partial (Q(\bar{I})Q(I(\mathbf{r})))}{\partial \mathbf{r}} = Q(\bar{I}) \frac{\partial Q(I(\mathbf{r}))}{\partial \mathbf{r}} + Q(I(\mathbf{r})) \frac{\partial Q(\bar{I})}{\partial \mathbf{r}}. \qquad (3)$$

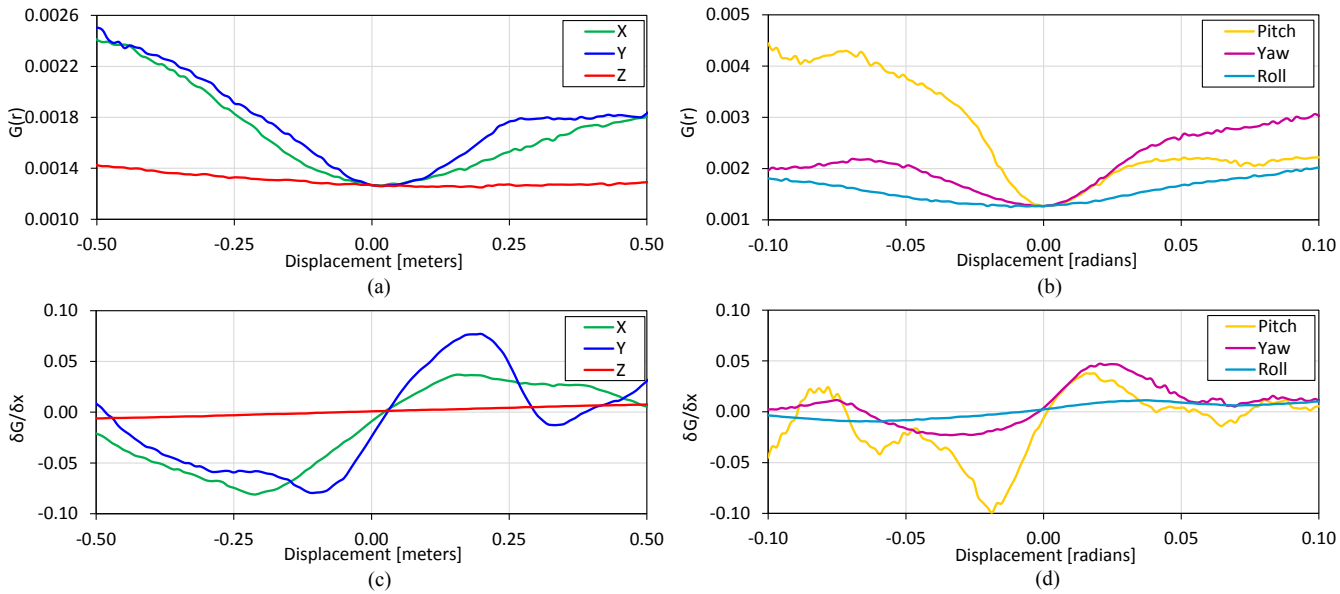

Fig. 4. The objective function of a sample image over (a) translational, and (b) rotational displacements of the rendered image camera position. The respective derived partial derivatives are shown in (c) and (d).

Here we note that the right-hand side of the sum in Eq. (3) can be removed, since the virtual camera parameters have no affect on $Q(\bar{I})$. Proceeding with the chain rule:

$$\frac{\partial(Q(\bar{I})Q(I(\mathbf{r})))}{\partial \mathbf{r}} = Q(\bar{I})\frac{\partial Q(I(\mathbf{r}))}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial \mathbf{r}}. \tag{4}$$

Now we can determine $\frac{\partial Q(I(\mathbf{r}))}{\partial \mathbf{x}}$, which is the gradient image of $Q(I(\mathbf{r}))$, and $\frac{\partial \mathbf{x}}{\partial \mathbf{r}}$, which is the image plane velocity of a pixel relative to the velocity of camera motion. This is commonly used in visual servoing applications and is known as the *Image Jacobian*. We present it here, and leave the derivation to the relevant literature [21]:

$$\frac{\partial \mathbf{x}}{\partial \mathbf{r}} = \begin{bmatrix} -\frac{\lambda}{Z} & 0 & \frac{u}{Z} & \frac{uv}{\lambda} & \frac{-\lambda^2-u^2}{\lambda} & v \\ 0 & -\frac{\lambda}{Z} & \frac{v}{Z} & \frac{\lambda^2+v^2}{\lambda} & -\frac{uv}{\lambda} & -u \end{bmatrix}, \tag{5}$$

where λ is the focal length of the camera in pixels. Note that Eq. (5) contains Z, which is supplied directly for each pixel from the rendering of $I(\mathbf{r})$ as a depth image. In the rendering process, the depth is provided within the vertex shader and passed to the fragment shader. An alternative would be to use the Z-buffer supplied by OpenGL, and replace λ in Eq. (5) with 1.0 since the Z-buffer is in normalized virtual camera space. In either case, we will face a problem using the depth image to retrieve Z values for each pixel, as we have performed a Gaussian convolution on the edge images. Therefore, the non-negative pixel values of $Q(I(\mathbf{r}))$ bleed past the areas that we have Z values for, resulting in an offset being introduced to the objective function derivatives. We overcome this by recognizing that convolved pixel areas of $Q(I(\mathbf{r}))$ are at the same depth as the edge itself. Therefore we can simply dilate the depth image $I(\mathbf{r})$ using the same kernel size as the Gaussian kernel used by Q, and use the dilated version to determine each pixel's Z value.

We can now determine the Jacobian matrix of the objective function, $\mathbf{J}$ by summing over all pixels and applying the reciprocal rule to the complete equation, giving:

$$\mathbf{J} = \left(\sum_{\mathbf{x}\in\bar{I},I(\mathbf{r})} Q(\bar{I})\frac{\partial Q(I(\mathbf{r}))}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial \mathbf{r}} \right) G(\mathbf{r})^2. \tag{6}$$

This is in a form that can be directly used by a gradient-based optimizer. We calculate approximations of the gradient images $\frac{\partial Q(I(\mathbf{r}))}{\partial \mathbf{x}}$ using the Scharr operator [22]. Fig. 4 (c) and (d) show the objective function derivatives for a sample image. The graphs show that all of the derived partial derivatives clearly pass through zero around the ground-truth pose, providing a stable optimization pathway. One point to note is that the Z direction partial derivative and cost function are the least sharply defined. This is the direction of the camera's optical axis, and when we consider typical highway scenes, the reason for this becomes apparent. Most of the scene structure from the voxel map is distant, so transformation in the Z direction does not significantly alter the rendered scene. Closer areas, such as lane markers and road boundaries, are usually parallel with the direction of motion. This would be less of an issue for city areas, where there are usually more structures close to the camera. In the highway environment of the HERE dataset, it causes a low longitudinal localization accuracy which must be compensated for by using odometry data to supply translation distance.

C. Localization Procedure

The position estimates of the optimization process are used as inputs to a Bayesian localization framework. For the first input image, position is initialized using the GPS receiver data supplied with the images. The camera yaw, pitch, and

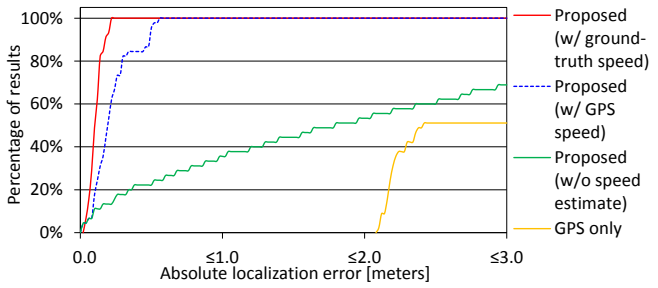

Fig. 5. Localization error of the tested systems, displayed as the percentage of results below each error threshold.

roll are initialized by finding the road-lane direction at the estimated camera position, based on the road-lane marker data in the map. All six camera parameters are incorporated into a second-order Kalman filter. With each new input image, the prediction stage of the filter estimates the next values of the camera parameters, which are then used as initial values for the visual optimization described above in Section IV-B. The covariances estimated by the Kalman filter are used to construct a bounding region two standard deviations wide for parameter optimization. The output of the optimization step provides the measurement update stage of the filter, with covariance estimates supplied by the Ceres solver. The filter then returns state estimates for all camera parameters of the input image.

The shallow cost function basin of the objective function of Eq. (2) in the Z direction provides a challenge for the optimizer, causing substantial drift in the position estimate along the direction of travel. We include an estimate of the scalar vehicle speed into the measurement update to overcome this. The speed estimate does not have to be highly accurate. A consumer grade GPS, or odometry hardware is sufficient to reduce drift. While the measurement noise of GPS speed estimates can generally be approximated by a normal distribution, the noise distribution of the sensor used would have to be considered for use with the Kalman filter.

V. Evaluation

To test the effectiveness of the proposed method, we performed localization of input images along using a 300 m section of the HERE dataset. The ground-truth data only provides capture locations, so we were unable to verify the optimization of Euler angles in the pose estimates except by visual inspection of each frame. However, the location of the camera is of interest, and angular pose estimation is only necessary for the optimization process.

We performed the localization process with a variety of vehicle speed inputs:

- No speed estimate input. Localization was performed with visual updates only.
- Vehicle speed estimated from GPS receiver.
- Vehicle speed estimated from the ground-truth positions with some normally distributed noise added, to simulate the kind of data that would be obtained from a normal vehicle speedometer.

TABLE I
Localization results.

Method	Avg. abs. error [m]	Standard deviation [m]	Max. error [m]
Filtered GPS only*	5.13	3.05	9.05
Proposed, no speed estimate	2.08	1.65	5.17
Proposed, GPS speed estimate	0.22	0.14	0.55
Proposed, ground-truth speed estimate†	0.11	0.05	0.21

*Consumer GPS readings filtered with a Kalman filter
†Ground-truth data used to simulate typical odometry sensor accuracy

The scalar speed value from the GPS data was calculated by averaging inaccurate GPS point motion. If a GPS satellite speed reading is available, this would be a much more suitable input, as GPS speed accuracy tends to be an order of magnitude superior to positional readings, even for inexpensive receivers [23]. The vehicle speed calculated from ground-truth data is included to indicate the potential level of localization accuracy when using a more accurate speed estimate, for example from wheel sensors.

The localization results are shown in Fig. 5 and Table I. Visual inspection of the camera images and corresponding rendered views shows that the majority of the localization error comes from longitudinal position estimate errors, and lateral alignment is very effective. Even when incorporating speed, longitudinal drift occurs if there are limited close-by road structures. Although not used in this experiment, the coarse GPS position measurements supplied in the dataset may also be able to be incorporated to reduce drift; however, because of the large biases observed (more than 30 m in places), careful application is necessary.

VI. Discussion

In this section we discuss the performance of the proposed method and identify areas for improvement.

The proposed system could resolve the viewing direction of the camera and lateral offset within the road-lane very well, but struggled with fine-grained longitudinal positioning in areas of the map where image edges were mostly parallel to the direction of motion. Using the localization method without a vehicle speed estimate resulted in the accumulation of longitudinal drift. Although Fig. 5 and Table I show that best localization accuracy was achieved when using a more precise vehicle speed estimate, even a rough estimate generated from noisy GPS receiver readings was sufficient to largely overcome this problem.

While optimization performance was surprisingly good given the limited information provided by the voxel renders, there are a few issues with the method. Initialization and parameter filtering before optimization are very important, as local minima are frequent. While convergence into local minima can be reduced by maintaining filtered parameter estimates, as we described in Section V, drift is still an

apparent problem. Long sections of self-similar road may cause longitudinal drift which is difficult to recover from. Combining with other sensors such as GPS may be the only solution to this issue when using such voxel maps. In addition, localization performance could likely be improved by combining optimization over multiple frames [24].

One important aspect of the localization method which was not considered in this research is the run-time. The current implementation as tested is not real-time, with the optimization for each input image taking in the order of 10 sec./frame on our test machine. We used a standard desktop computer running Linux with an Ivybridge i7 Intel processor and 16 GB of RAM, with on-board graphics. Our implementation included no particular multi-threading or optimization, as it is presented here as a proof of concept rather than a complete solution. Based on the performance of more complex direct localization methods, real-time localization should be achievable with optimization and more powerful dedicated graphics hardware. The main performance bottleneck is in the Jacobian calculations, taking on average 80% of the optimization time. The Jacobian calculations were completely performed on the CPU. The other major computational bottleneck is the voxel map image rendering process, which again would be significantly speeded-up by using a dedicated GPU.

The map used by the method described in this paper is very compact. The voxel data, stored in JSON format, occupies less than 180 MB/km. Dense feature point maps or textured 3D maps contain much more data so are more difficult to store or stream for expansive environments.

VII. Conclusion

In this paper, we proposed an automotive localization method which employs a lightweight voxel map for direct visual camera pose estimation. The objective function for use within a non-linear optimizer leverages mutual edge areas of images from the rendered voxel map and in-vehicle camera images. Derivations for the analytical derivatives of the objective function were explored for efficient optimization with the Levenberg-Marquardt method.

The results of the proposed method show that even with the relatively small amounts of information contained in a coarse voxel map, direct visual localization is not only possible but accuracy levels that could be sufficient for many automated driving tasks are achievable. We plan to further this work to enable real-time operation through optimized implementation and hardware, and also investigate methods to provide robust localization over long stretches of road without drift.

Acknowledgment

Parts of this research were supported by MEXT, Grant-in-Aid for Scientific Research.

References

[1] H. Lategahn and C. Stiller, "Vision-only localization," *IEEE Trans. Intelligent Transportation Systems*, vol. 15, no. 3, pp. 1246–1257, June 2014.

[2] D. Wong, D. Deguchi, I. Ide, and H. Murase, "Position interpolation using feature point scale for decimeter visual localization," in *Proc. 2015 IEEE Int. Conf. on Computer Vision (ICCV2015) Workshops, Santiago, Chile*, Dec. 2015, pp. 90–97.

[3] H. Badino, D. F. Huber, and T. Kanade, "Real-time topometric localization," in *Proc. 2012 IEEE Int. Conf. on Robotics and Automation (ICRA2012), St. Paul, MN, USA*, May 2012, pp. 1635–1642.

[4] M. Milford, "Visual route recognition with a handful of bits," in *Proc. 2012 Robotics: Science and Systems Conf., Sydney, Australia*, July 2012, pp. 297–304.

[5] G. Pascoe, W. Maddern, and P. Newman, "Robust direct visual localisation using normalised information distance," in *Proc. 26th British Machine Vision Conference (BMVC2015), Swansea, Wales, UK*, vol. 3, Sept. 2015, pp. 70.1–70.13.

[6] R. W. Wolcott and R. M. Eustice, "Visual localization within LIDAR maps for automated urban driving," in *Proc. IEEE/RSJ 2014 Int. Conf. on Intelligent Robots and Systems (IROS2014), Chicago, IL, USA*, Sept. 2014, pp. 176–183.

[7] G. Caron, A. Dame, and E. Marchand, "Direct model based visual tracking and pose estimation using mutual information," *Image and Vision Computing*, vol. 32, no. 1, pp. 54–63, Jan. 2014.

[8] D. Deguchi, K. Mori, M. Feuerstein, T. Kitasaka, C. R. Maurer, Y. Suenaga, H. Takabatake, M. Mori, and H. Natori, "Selective image similarity measure for bronchoscope tracking based on image registration," *Medical Image Analysis*, vol. 13, no. 4, pp. 621–633, Aug. 2009.

[9] HERE. (Accessed: 2016-11-29) HERE. [Online]. Available: https://here.com/en

[10] J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," *Numerical analysis*, pp. 105–116, 1978.

[11] H. F. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robotics and Automation Mag.*, vol. 13, no. 2, pp. 99–110, June 2006.

[12] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[13] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[14] H. Uchiyama, D. Deguchi, T. Takahashi, I. Ide, and H. Murase, "Ego-localization using streetscape image sequences from in-vehicle cameras," in *Proc. 2009 IEEE Intelligent Vehicles Symposium (IV2009), Xi'an, China*, June 2009, pp. 185–190.

[15] H. Kume, A. Suppé, and T. Kanade, "Vehicle localization along a previously driven route using image database," in *Proc. 13th IAPR Conf. on Machine Vision Applications (MVA2013), Kyoto, Japan*, May 2013, pp. 177–180.

[16] S. Baker and I. Matthews, "Lucas-Kanade 20 years on: A unifying framework," *Int. J. Computer Vision*, vol. 56, no. 3, pp. 221–255, Feb. 2004.

[17] P. Viola and W. M. Wells III, "Alignment by maximization of mutual information," *Int. J. Computer Vision*, vol. 24, no. 2, pp. 137–154, Sept. 1997.

[18] The University of Melbourne. (Accessed: 2016-11-29) ITS World Congress 2016 Grand Challenge. [Online]. Available: http://conference.eng.unimelb.edu.au/its-gc/

[19] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, Nov. 1986.

[20] S. Agarwal, K. Mierle, and Others. (Accessed: 2016-11-29) Ceres solver. [Online]. Available: http://ceres-solver.org/

[21] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Trans. Robotics and Automation*, vol. 12, no. 5, pp. 651–670, Oct. 1996.

[22] H. Scharr, "Optimal filters for extended optical flow," in *Complex Motion —IWCM 2004 First Int. Workshop, Gnzburg, Germany*, ser. Lecture Notes on Computer Science, Oct. 2004, vol. 3417, pp. 14–29.

[23] M. Modsching, R. Kramer, and K. ten Hagen, "Field trial on GPS accuracy in a medium size city: The influence of built-up," in *Proc. 3rd Workshop on Positioning, Navigation and Communication (WPNC2006), Hannover, Germany*, March 2006, pp. 209–218.

[24] G. Pascoe, W. Maddern, and P. Newman, "Direct visual localisation and calibration for road vehicles in changing city environments," in *Proc. 2015 IEEE Int. Conf. on Computer Vision (ICCV2015) Workshops, Santiago, Chile*, Dec. 2015, pp. 98–105.