Online Database Updating by Change Detection

Philippe Simard and Frank P. Ferrie

McGill University, Montreal, Quebec, Canada

ABSTRACT

Synthetic vision systems render artificial images of the world based on a database and position/attitude information of the aircraft. Due to both its static nature and inherent modelling errors, the database introduces anomalies in the synthetic imagery. Since it reflects at best a nominal state of the environment, it often requires updating via online measurements. The latter can vary from correction of pose and geometry to more complex operations such as marking the locations of detected obstacles. This paper presents an approach for detecting database geometric anomalies online. Since range sensors have a low update rate, they cannot be used for quick validation. Instead of range data, the proposed technique employs an imaging sensor, which can be of any type. It takes advantage of the fact that given a geometric model of the scene and known motion of the observer, the sensor image warping can be exactly predicted. If the geometry of the database is incorrect, the sensor image will not be correctly predicted and geometric differences will thus be detected. The algorithm is tested against simulated imagery and results show that it can correctly identify geometric anomalies. It can cope with known misalignment of the database and pose estimation errors. The technique is shown to be quite robust in low visibility conditions, given that the validated features are at least partially visible. It also automatically detects if a motion is sufficient for a given sensitivity.

Keywords: database updating, change detection, model validation, range sensor

1. INTRODUCTION

In aviation applications, synthetic vision systems create an artificial view of the world from two sources of information: a database built offline and the position/attitude information of the aircraft. Although the precision of the synthetic imagery is dependent on the quality of both, the largest source of error is generally the database. It is built often long before usage and therefore reflects a past version of the world. For example, structures such as buildings could have been introduced or removed. Furthermore, the modeling process itself introduces errors in the database. Because of different compromises for real-time rendering, certain objects/features could have been omitted or the resolution of the database lowered. The precision of the source data also adds errors to the database. For all these reasons, online database updating would definitely increase the performance of synthetic vision systems.

In this paper, a technique is proposed for automatically and quickly detecting geometric anomalies in a database. These anomalies can be of any type, provided they result in an incorrect geometry of the world. The approach requires any type of imaging device, such as standard cameras or infrared sensors, as long as its projective geometry is known. The technique is differential and thus, requires movement of the sensor. It uses the geometric information of the database to predict how the sensor image should change. If the measured change does not match the predicted change, the database is assumed to be incoherent with the real world. The algorithm produces a two-dimensional change map that shows locations where a geometric anomaly was detected. This change map could further be used to either drive a low update rate range sensor or to graphically code the database according to its correctness.

2. BACKGROUND

This work was performed as part of a program with CAE Inc., the National Research Council (NRC) of Canada, BAE Systems Canada, the University of Toronto, York University, and the Department of National Defence (Canada) to develop an Enhanced and Synthetic Vision System (ESVS) for low-visibility flight in the Search and Rescue application[4]. The ESVS includes an on-board image generator to render synthetic images of the terrain and an infrared sensor to provide a correlated out-the-window image. Both images are presented on a helmet-mounted-display with the infrared image fused as an inset in the center of the synthetic image field of view[9,10]. Flight symbology is superimposed on the final image to assist the pilot in flying the aircraft. An active range sensor will also be used to correct database inaccuracies due to erroneous or incomplete source data. The ESVS display concept is presented in Figure 1.

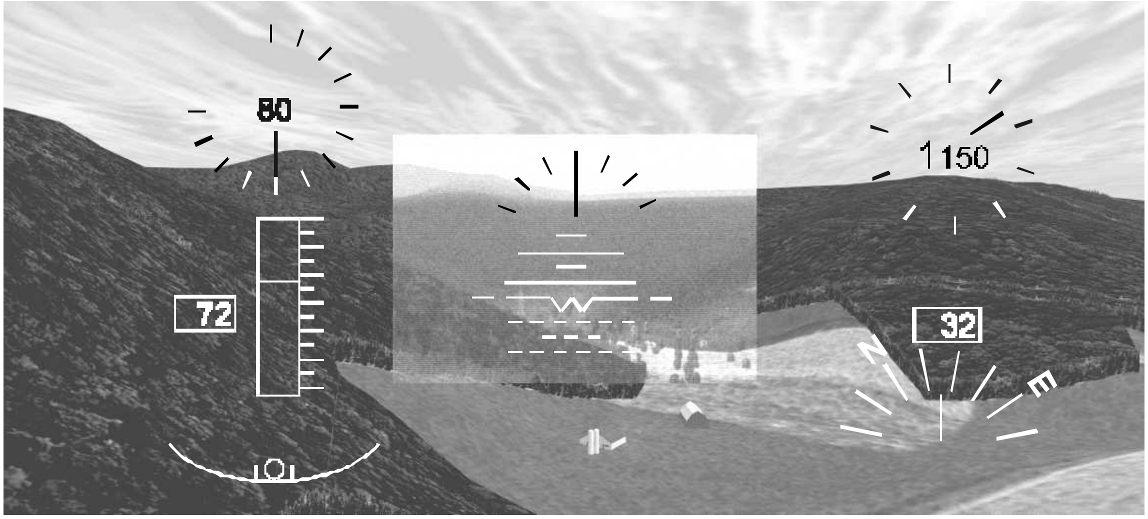

Figure 1. ESVS concept

Because range sensors have low update rates, they cannot be used to scan the entire scene in real-time. Thus, the scanning strategy has to be optimized to make good use of the ranging device. If database anomalies can be quickly detected, a change map could be used to drive the range sensor toward those locations. Previous work has been done toward this goal. Nevatia et al.[1,3] have developed a technique to confirm the presence of man-made structures (buildings) in aerial images. Watanabe, Miyajima and Mukawa[11] have developed an algorithm to validate the presence and geometry of buildings, using a shadow and shading model. Leclerc et al.[6,7,8] proposed a method that uses aerial images for detecting changes in shapes that can be modelled as single-valued functions. Zha, Tanaka and Hasegawa[12] have developed a technique for updating two-dimensional maps used by mobile robots.

Unfortunately, these methods can only deal with man-made structures[1,3,11], simple shapes[6,7,8] or two-dimensional maps[12]. They cannot validate the geometry of complex databases such as those required in synthetic vision systems. The proposed technique can deal with any three-dimensional geometric model, which can be of any complexity.

In order to find locations where the geometry of the world is incorrect, the imaging data must be compared with the three-dimensional model (database). A priori, the comparison can be done in either two or three dimensions. If one chooses to perform it in the imaging space, a synthetic image has to be generated from the database and compared with the imaging data. In such a case, a realistic synthetic image must be rendered. This is very hard, as the exact image generation parameters have to be known (lighting conditions, ground textures, time of day, season, etc.). One can choose, on the other hand, to perform the comparison in three dimensions. The geometry of the world has then to be extracted from the imaging data and compared against the database. This recovery operation amounts to solving the computer vision problem in a natural scene domain, which is a significantly difficult problem.

The proposed technique takes advantage of both sources of information, and avoids the problems of choosing one comparison space as described above. It is based on optical flow and uses the scene model to predict the next position of pixels in the current sensor image, given some induced movement (this is referred as image based rendering[2,5]). A predicted sensor image is rendered according to the flow prediction, and then compared with the observed sensor image. If the geometry of the model is incorrect, the predicted image will be distorted and will not match the observed one. This results in a change map that displays locations for which the model is incorrect.

3. CHANGE DETECTION

3.1 Predicted Image Rendering

In order to derive the optical flow equations necessary for the predicted image generation, a viewer-based coordinate system is adopted (Figure 2). Note that points in the camera's image plane are denoted using lower case characters. The origin is at the focal point of the camera and the image plane is at Z= *f*. The Z-axis runs along the optical axis, and the X- and Y-axes are parallel to the x- and y-axes of the image plane respectively.

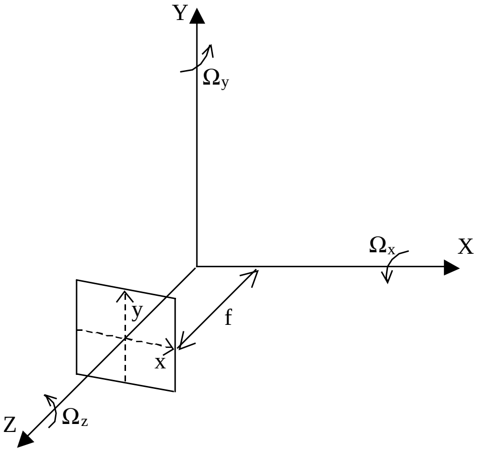

Figure 2. Viewer-centered coordinate system

The ego-motion of the camera is decomposed into a rotation about an axis passing through the origin, and a translation. Any three-dimensional motion can be represented as such. This is denoted as (T°R), where ° indicates a composite function. Given a point in three-dimensional space, (X_i,Y_i,Z_i), and a camera motion, $(T_x,T_y,T_z)°(R_x,R_y,R_z)$, the new location of the point, $(X_{i+1},Y_{i+1},Z_{i+1})$, is given by:

$$\begin{bmatrix} X_{i+1} \\ Y_{i+1} \\ Z_{i+1} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & -\Omega_z & \Omega_y & T_x \\ \Omega_z & 1 & -\Omega_x & T_y \\ -\Omega_y & \Omega_x & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}. \qquad (1)$$

This homogenous operator assumes the small rotation approximation, $sin\theta = \theta$. A second system of equations provides a perspective projection model for the pinhole camera with focal length *f*,

$$\begin{bmatrix} x_i \\ y_i \\ f \\ f/Z_i \end{bmatrix} = \begin{bmatrix} f/Z_i & 0 & 0 & 0 \\ 0 & f/Z_i & 0 & 0 \\ 0 & 0 & f/Z_i & 0 \\ 0 & 0 & 0 & f/Z_i \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}. \qquad (2)$$

From (1) and (2), a system of equations for the new location, (x_{i+1}, y_{i+1}) of a point in the image (x_i, y_i), after the viewpoint has moved can be derived:

$$x_{i+1} = \frac{f A_i}{C_i},$$

$$y_{i+1} = \frac{f B_i}{C_i}, \qquad (3)$$

where,

$$A_i = x_i + \Omega_z y_i + f\,\Omega_y + \frac{T_x f}{Z_i},$$

$$B_i = y_i + \Omega_z x_i - f\,\Omega_x + \frac{T_y f}{Z_i},$$

$$C_i = \Omega_x y_i - \Omega_y x_i + f + \frac{T_z f}{Z_i}. \tag{4}$$

Using (3) and (4), the predicted sensor image is constructed by moving each pixel of the original sensor image. In other words, the original image is warped according to the geometric information of the database and to the induced motion. Disocclusions (locations in the world that are not visible from the original viewpoint and which become visible from the new viewpoint) will result in unpredictable pixels in the predicted image and are flagged as such. These pixels are omitted from the comparison step, as it will be explained in the next sub-section.

3.2 Change Map

The predicted image is compared with the observed image using a region-based approach. The metric used is the sum-of-absolute differences (*SAD*). It is computed for each location of the image over a neighborhood of size *2n+1*:

$$SAD = \sum_{l=-n}^{l=n} \sum_{k=-n}^{k=n} \left| P(\mathbf{x}_{i+1} + (k,l)) - O(\mathbf{x}_{i+1} + (k,l)) \right|, \tag{5}$$

where *P* is the predicted intensity value, *O* the observed intensity value and $\mathbf{x}_{i+1}$ is the vector (x_{i+1},y_{i+1}). The *SAD* will be large if the two regions are different, meaning that the geometry of the model is incorrect for that particular location. Conversely, if it is small, the geometry is assumed to be correct. These conclusions can be made under the assumption that neighboring pixels are the projection of neighboring 3D points, which is in general the case. Also note that the *SAD* is computed only in neighborhoods where the predicted image is computed, i.e. where there is no disocclusion.

3.3 Database Misalignment and Pose Errors

Because of the finite precision of pose sensors, there will be errors in the measured position and attitude of the viewer. Global alignment errors of the database are also expected. These types of errors will result in incorrect values for x_{i+1} and y_{i+1} and will cause the algorithm to detect anomalies when it should not. To account for these problems, the maximum induced error in x_{i+1} and y_{i+1} is calculated:

$$\Delta x_{i+1} = \frac{\partial x_{i+1}}{\partial T_x}\Delta T_x + \frac{\partial x_{i+1}}{\partial T_y}\Delta T_y + \frac{\partial x_{i+1}}{\partial T_z}\Delta T_z + \frac{\partial x_{i+1}}{\partial \Omega_x}\Delta\Omega_x + \frac{\partial x_{i+1}}{\partial \Omega_y}\Delta\Omega_y + \frac{\partial x_{i+1}}{\partial \Omega_z}\Delta\Omega_z,$$

$$\Delta y_{i+1} = \frac{\partial y_{i+1}}{\partial T_x}\Delta T_x + \frac{\partial y_{i+1}}{\partial T_y}\Delta T_y + \frac{\partial y_{i+1}}{\partial T_z}\Delta T_z + \frac{\partial y_{i+1}}{\partial \Omega_x}\Delta\Omega_x + \frac{\partial y_{i+1}}{\partial \Omega_y}\Delta\Omega_y + \frac{\partial y_{i+1}}{\partial \Omega_z}\Delta\Omega_z, \tag{6}$$

where ΔT_x, ΔT_y, ΔT_z, $\Delta\Omega_x$, $\Delta\Omega_y$ and $\Delta\Omega_z$ are the combined errors in alignment and pose estimate. The minimum *SAD* in the circular neighborhood described by the vector $(\Delta x_{i+1}, \Delta y_{i+1})$ is then used for the change map generation. This allows the technique to deal with any range of database misalignment and pose uncertainties. However, as their value increases, the number of operations required to compute the change also increases.

3.4 Minimum Motion

Although any motion induced to the camera will produce image flow, translations will be the only ones that induce change detection. The reason is that the image flow does not depend on the depth values of the scene in the case of rotations about the focal point (in the viewer-centered coordinate system shown in Figure 2). Thus, the viewer's motion must contain at least a translation (in any direction) and must be large enough to induce sufficient image flow. Because of high refresh rates of imaging devices, the motion between two successive frames could be too small. An appropriate strategy is required to decide whether the displacement is large enough.

A displacement is sufficient if a given depth difference between the world and the model can be detected. Such a difference in depth, given some motion of the observer, leads to a displacement in the image plane expressed by:

$$\Delta xd_{i+1} = \frac{\partial x_{i+1}}{\partial Z_i} d \ ,$$
$$\Delta yd_{i+1} = \frac{\partial y_{i+1}}{\partial Z_i} d \ , \tag{7}$$

where *d* is the difference in depth. If the desired minimum detectable difference is *d*, the pixel displacement should be greater than the tolerable displacement due to database misalignment and pose errors:

$$\left\| \mathbf{\Delta xd}_{i+1} \right\| > \left\| \mathbf{\Delta x}_{i+1} \right\| , \tag{8}$$

where $\mathbf{\Delta xd}_{i+1}$ is the vector $(\Delta xd_{i+1}, \Delta yd_{i+1})$. Note that $\mathbf{\Delta xd}_{i+1}$ is computed only with the translation components of the movement, since rotations do not allow for change detection. If that condition holds for a great part of the image, the motion is considered large enough and the change map is calculated. In theory, *d* should be set as close to zero as possible but in practice, doing so could result in a required motion that would be too large to have reasonable overlap between the frames.

4. RESULTS

4.1 The Simulation Environment

A simulator was developed to test the change detection algorithm. It was implemented in C++ using the OpenGL rendering package. The system renders realistic images to simulate standard camera imagery. The imaged world consists of a hilly terrain with forested regions and a few buildings. A simulated sensor image is shown in Figure 3. The synthetic database contains a terrain with decreased resolution (160m terrain elevation posts) in order to simulate synthetic database modelling compromises for real-time rendering. A sample view of the database is presented in Figure 4. Note that the ridge in the left part has been lowered by 50m to simulate local terrain elevation errors that could be found in digital elevation maps. A good change detection algorithm should thus be able to detect that the synthetic database is missing the buildings, and that the geometry of the ridge is incorrect.

4.2 Basic Test

The algorithm was first tested by moving the observer to the left by 100m, with the starting viewpoint being the one used for generating Figure 3 and Figure 4. The simulated sensor image and the predicted sensor image after inducing the motion are shown in Figure 5 and Figure 6 respectively. Observe that the sensor image was correctly predicted in locations where the geometry of the database was correct. However, the appearance of the silo, the barn and the ridge are distorted. In particular, the silo is tilted, although its base is in the correct position. The reason is that the base of the silo is lying on the ground, which is geometrically correct in the database.

The corresponding change map is shown in Figure 7. Note that locations where the algorithm has detected changes are shown in light tones. The ridge and the silo where correctly identified as missing from the database. The barn produced a weaker response, mainly because it does not reflect a large discrepancy in depth between the database and the real world from the given viewpoint. Some anomalies in the far ridges were also detected due to the coarse resolution of the synthetic database. Because these ridges are the furthest visible features, even slight differences in their heights result in a large difference in depth.

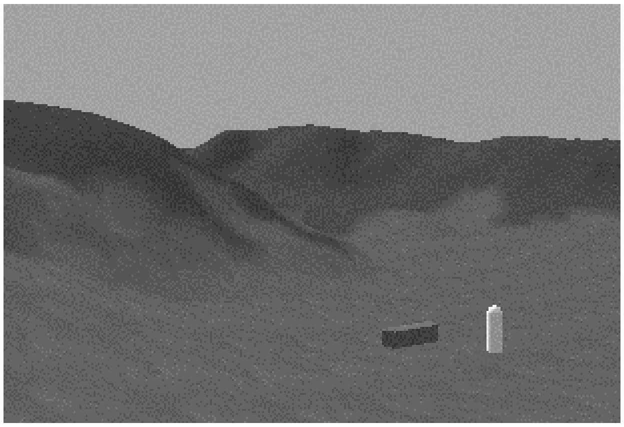

Figure 3. Simulated sensor image

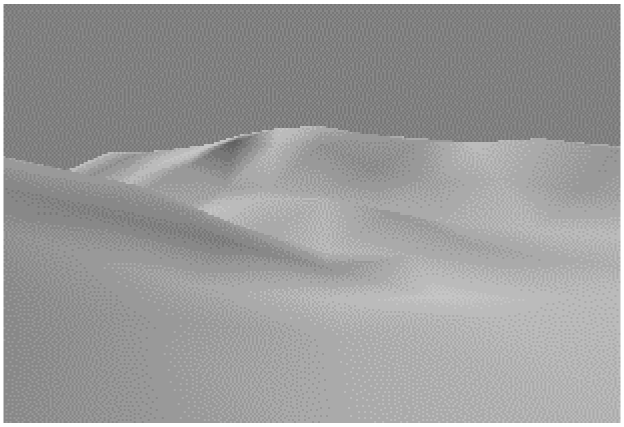

Figure 4. Database view

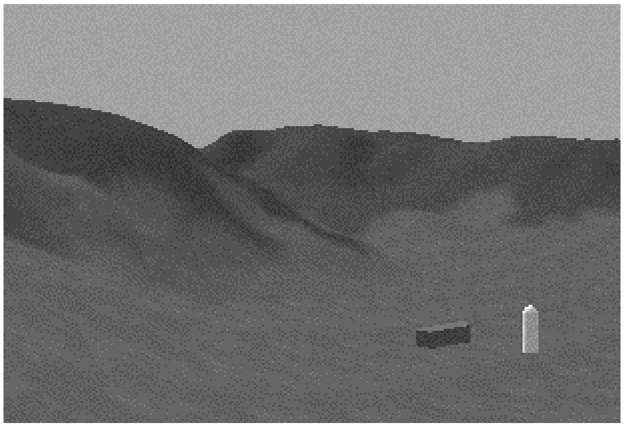

Figure 5. Simulated sensor image after left motion

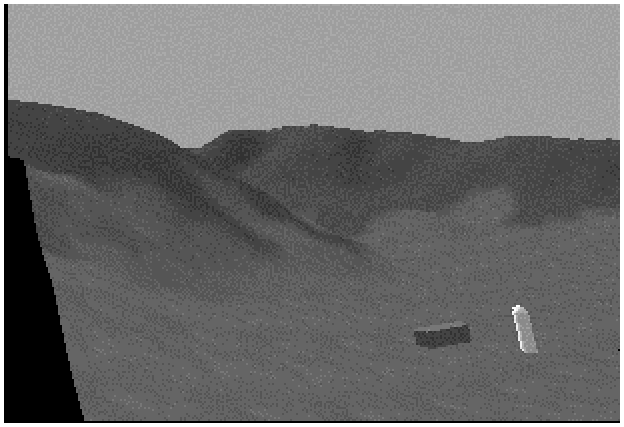

Figure 6. Predicted image

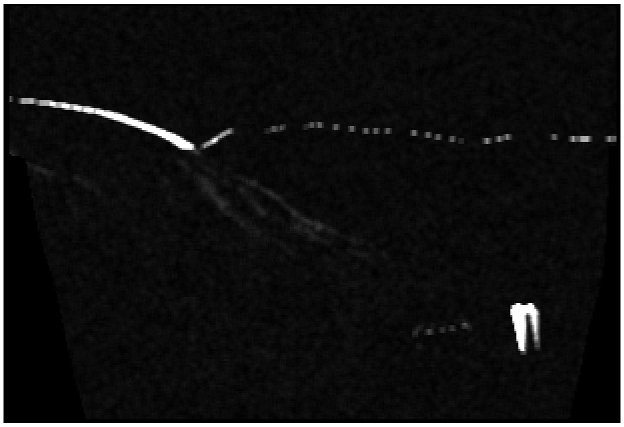

Figure 7. Change map

4.3 Effects of Database Misalignment and Pose Errors

As outlined earlier, database misalignment (due to errors in source data) and pose estimation errors could cause erroneous detected anomalies. Alignment errors in translation (50m in all three directions) and rotation (1^o around each axis) were thus introduced to simulate the expected discrepancy. Position and attitude errors of 1m (in latitude, longitude and altitude) and 1^o (in roll, pitch and yaw) were also added to reflect positional and attitude estimation errors. Figure 8 shows the change map produced when all of these errors are present, but when the tolerance technique described in Section 3.3 is not applied. Note that an upward motion of 20m was induced. Observe that the ridge, the silo and the barn are still detected but that their positions are less precisely outlined. In addition, large changes were detected in the far ridges. These should be in fact tolerated and should not show up in the change map, as they are due to the known database misalignment and pose errors. The change map shown in Figure 9 presents the results obtained by applying the error tolerance technique. The ridge error and the silo are still detected but the far ridges and the barn do not show up anymore. The technique can thus tolerate database misalignment and pose errors, but the larger they are, the less sensitive the algorithm becomes.

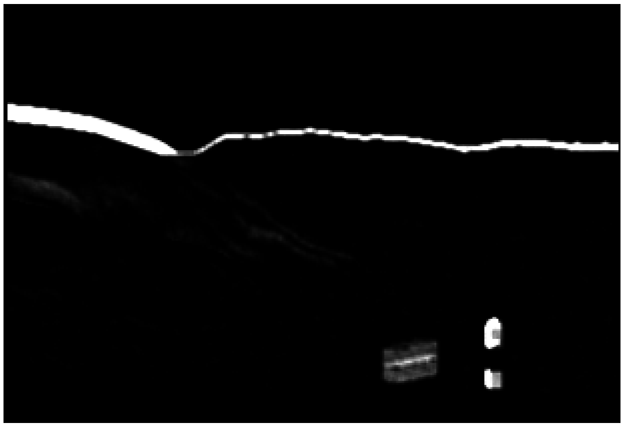

Figure 8. Change map without error tolerance

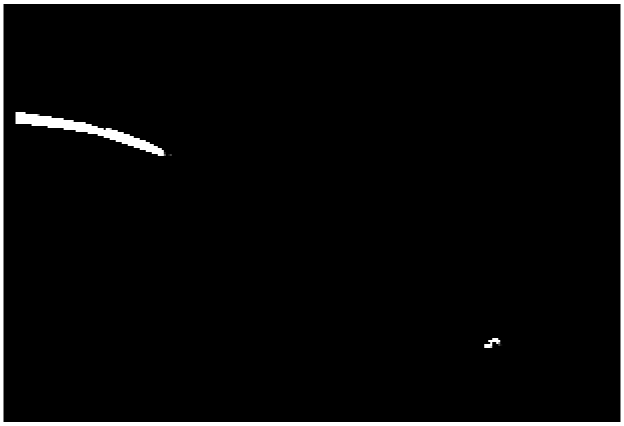

Figure 9. Change map with error tolerance

4.4 Performance in Low Visibility

Since the technique should also work in poor visibility conditions, some fog was added in the sensor simulation. The visibility was set to about 1km and the corresponding sensor image is shown in Figure 10. A forward motion of 100m was then induced to the observer, and the corresponding change map is depicted in Figure 11. The ridge and the silo are still flagged as changes and surprisingly, the barn also shows up. This is due to the fact that the sensitivity was greater for that location, given the forward motion. The algorithm thus has the ability of performing well even in poor visibility conditions, given that features are at least partially visible in the sensor image.

Figure 10. Simulated sensor image with fog

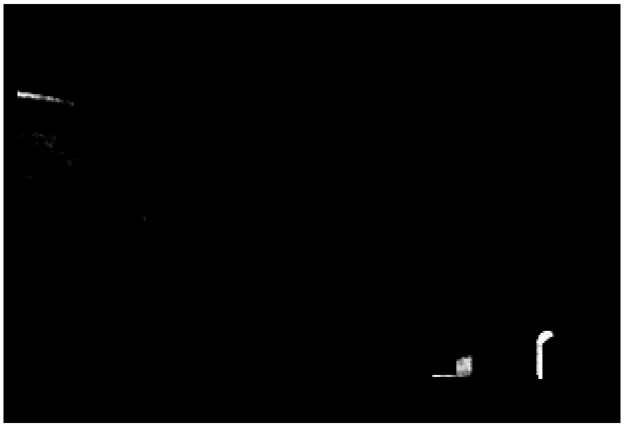

Figure 11. Change map

4.5 Motion vs. Sensitivity

As previously stated, any motion having a translation component will induce change detection. However, the motion should be large enough such that a reasonable anomaly in depth can be detected over a sufficient portion of the image. Figure 12 shows the sensitivity map for a desired detectable change of at least 50m given a left displacement of 20m. The dark regions indicate locations where the condition defined by (8) was not met. Conversely, the light region represents locations where at least a 50m geometric difference can be detected. In fact, the closer the features are, the easier it is to detect an anomaly given some motion. For an upward displacement of 20m, the sensitivity map is exactly the same. However, in order to get a similar sensitivity map, the required forward motion has to be 100m (Figure 13). In this case, the ability to detect a given depth difference is dependent not only on the distance of the feature, but also on its position in the image: the sensitivity becomes smaller as the feature becomes closer to the image center.

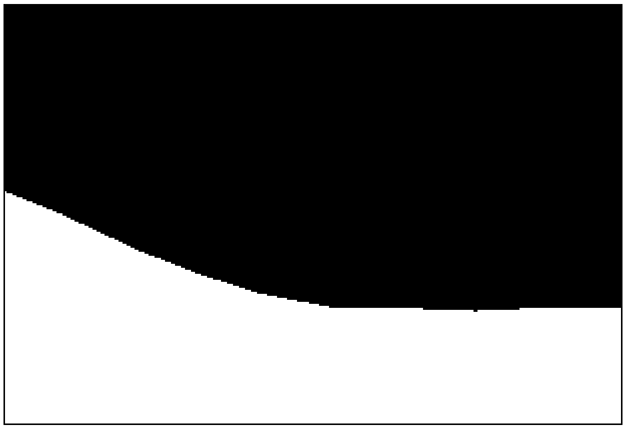

Figure 12. Sensitivity map (10m left/upward)

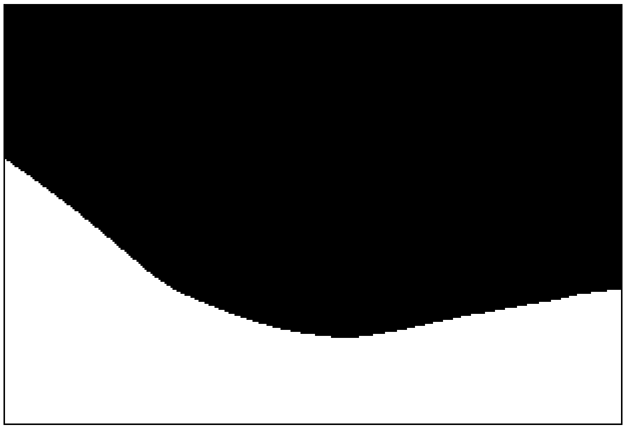

Figure 13. Sensitivity map (100m forward)

5. CONCLUSIONS

The results showed that geometric anomalies in synthetic databases can be detected, with an imaging sensor only. Hence, a change map could be calculated online, which would allow optimal use of a low-update rate range sensor. The algorithm can cope with known database misalignment and pose estimation errors. It is robust to noise and decreased visibility, as long as features are at least partially visible. Given a required sensitivity to depth differences, the technique can automatically determine if a given motion is large enough.

An initial system without ranging sensors is undergoing flight trials at the NRC Flight Research Laboratory near Ottawa in April 2001. Results to date are promising, and future work will concentrate on the integration of a range sensor with the system. The algorithm presented in this paper will be tested offline this year against real flight data obtained through the program team at the University of Toronto and NRC.

Future work will involve the development of an algorithm for using the change map to drive a range sensor. Different criteria, such as proximity or detected change size, will be investigated. Optimization of the dynamic properties of the sensor steering mechanism will also be considered. Finally, in order to integrate the detected changes with an online database for a standard PC-based image generator, algorithms will be developed to distinguish between changes in the underlying terrain and obstacles on the terrain.

6. ACKNOWLEDGMENTS

The authors would like to thank Marcel Mitran for deriving the optical flow equations.

7. REFERENCES

1. Bejanin, M., Huertas, A., Medioni, G., Nevatia, R., “Model Validation for Change Detection”, *Proceedings of the Second IEEE Workshop on Applications of Computer Vision*, pp. 160-167, 1994.
2. Debevec, P.E., Taylor, C.J., and Malik, J., “Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach”, *SIGGRAPH '96*, pp.11-20, 1996.
3. Huertas, A., Nevatia, R., “Detecting changes in aerial views of man-made structures”, *International Conference on Computer Vision*, pp. 73-80, 1998.
4. Kruk, R.V., Link, N.K., MacKay, W.J., Jennings, S., “Enhanced and Synthetic Vision System for Helicopter Search and Rescue Mission Support.” *Proc. American Helicopter Society 55th Annual Forum*, Montreal, Quebec (Canada), 1999.
5. Laveau, S., and Faugeras, O. “3-D Scene Representation as a Collection of Images”, *Proceedings of the International Conference on Pattern Recognition*, pp.689-691, 1994.
6. Leclerc, Y. G., “Continuous terrain modeling from image sequences with applications to change detection”, *Proceedings of the DARPA Image Understanding Workshop*, 1997.
7. Leclerc, Y. G., Luong , Q.-T., Fua, P., “A framework for detecting changes in terrain”, *Proceedings of the DARPA Image Understanding Workshop*, 1998.
8. Leclerc, Y. G., Luong , Q.-T., Fua, P., “Detecting changes in 3-d shape using self-consistency”, *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2000.
9. Simard, P., Link, N.K., Kruk, R.V., “Evaluation of algorithms for fusing infrared and synthetic imagery”, *Proc. of the SPIE Conference on Enhanced and Synthetic Vision*, vol. 4023, pp. 127-138, 2000.
10. Simard, P., Link, N.K., Kruk, R.V., “Feature detection performance with fused synthetic and infrared imagery”, *Proc. Human Factors and Ergonomics Society 43rd Annual Meeting*, pp. 1108-1112, 1999.
11. Watanabe, S., Miyajima, K., Mukawa, N., “Detecting changes of buildings from aerial images using shadow and shading model”, *International Conference on Pattern Recognition*, pp. 1408-1412, 1998.
12. Zha, H., Tanaka, K., Hasegawa, T., “Detecting changes in a dynamic environment for updating its maps by using a mobile robot”, *IROS '97*, pp. 1729-1734, 1997.