Fast Realization of Digital Elevation Model

X. Descombes[1], A.I. Kravchonok[2], P.V. Lukashevich[2], B.A. Zalesky[2]
1) INRIA, 2004 route des Lucioles, BP 93, 06902, Sophia Antipolis Cedex, FRANCE, xavier.descombes@sophia.inria.fr
2) United Institute of Informatics Problems, Minsk, Surganova 6, Belarus, zalesky@newman.bas-net.by

Abstract: *We propose an optimization approach to speed up the point matching process underlying the 3D reconstruction of complex urban scenes. We consider the Optical Flow technique for point matching and propose to introduce MMX and SSE2 instructions to accelerate significantly the matching process. Fast point matching allows using sub-pixel image resolution, which provides a more accurate estimation of the Optical Flow by exploiting wider correlation windows, and therefore improves the final quality of urban scenes 3D reconstructions.*

Keywords**:** calibrated stereo pair, urban scenes, 3D reconstruction, MMX and SSE2 acceleration.

1. INTRODUCTION

With the new generation of high resolution optical images, the problem of 3D urban areas reconstruction from stereo pair of images appears to be challenging. It finds successful applications in many areas of human activity such as geography, architecture or information technologies.

Numerous approaches have been proposed to solve it [1, 2, 3, 4, 5]. Among them, the most popular consists in first computing a Digital Elevation Model (DEM) [3]. Computing this 3D model is equivalent to estimate the disparity map of corresponding pixels in the calibrated images [3, 6, 7]. Therefore, the main problem reduces to a matching problem between the two images.

The accuracy of the DEM depends on many factors; among them the image resolution is crucial. High resolution leads to images with a huge size inducing heavy computations for constructing the DEM. Several hours or even several days are sometimes required to obtain the solution. Thus, to speed-up the process appears to be a major issue.

In this context, different techniques have been proposed to optimize the computation time required for reconstructing a DEM [8]. We present a new approach to this problem that significantly speeds up the DEM computation. To compute a dense disparity map we exploit the Optical Flow information [7]. We select the region-based matching method, proposed in [9, 10], to compute the optical flow on the stereo pair. This choice allows an implementation which extensively uses Multimedia Extensions (MMX) and Streaming SIMD Extensions (SSE) that reduce the execution time by several orders of magnitude. The bulk of arithmetical operations, which actually are needed to compute the pixels matching in the stereo pair, are performed with the help of MMX or SSE2 instructions.

Optimizing the DEM reconstruction reduces the computations time but also improves the accuracy of the result as it allows using larger window for estimating the correlation between pairs of pixels.

Herein, we do not consider processing partly visible pixels and moving objects, although it can be done using the optical flow information. This aspect will be considered in future works.

The proposed approaches are tested numerically on a stereo pair of aerial images. The results show a real impact of the proposed techniques on both the accuracy of the reconstructed scenes and the reduction of execution time needed for their computation.

2. OPTICAL FLOW FOR URBAN DEM

Determining a dense disparity map, leading to a Digital Elevation Model, is a very computationally expensive task. Our goal is to develop a technique that drastically accelerates the process in order to reconstruct complex urban scenes. The optical flow approach is selected as the simplest technique allowing a huge optimization of the algorithm. Several optical flow methods have been proposed in the literature. They are based on differential equations [9, 11], region-based matching, energy minimization or phase estimation [9]. The region-based matching appears to be the simplest and less computationally demanding [10].

Before giving a detailed description of the optimization technique for the region-based matching, let us note a specificity of an urban DEM. Usually, stereo pairs of urban areas are obtained from a rather high altitude. Therefore, depth of urban scenes is much less than the distance from the earth surface to the optical center of the camera. In this case, the perspective effects can be neglected and we can consider an affine transform of the scene [12]. We can even often consider only translations, or in worst cases a composition of a translation and an affine scaling along coordinate axes. This specificity allows a preliminary alignment of the stereo pair, either by the appropriate shift, or by the combination of a shift and an affine scaling. For instance, the stereo images of the urban scene shown in Fig.1 and 2 can be aligned by simple shifts. After such an alignment, the maximum disparity between corresponding pixels in the 1^{st} and the 2^{nd} images is equal to 6 pixels. Therefore, we make the following assumption:

Assumption: Images of a urban scene stereo pair can be aligned either by a shift or the combination of a shift and an affine scaling.

Since, after the preliminary alignment, the maximum

Fig.1 – Left image of an urban scene calibrated stereo pair © French Geographic Institute (IGN).

Fig.2 – Right image of an urban scene calibrated stereo pair © French Geographic Institute (IGN).

of the disparity map is not large, we can avoid considering the epipolar geometry. Using epipolar lines does not increase the accuracy but can slow down seriously the computations.

The best matching between points is usually obtained by maximizing the correlation or minimizing the norm L^2 or L^1 between rectangular or square windows [13]. We consider, in this study, square windows and the norm L^1, more adapted to urban areas at high resolution, which contain high contrasts between objects, and allowing the most significant acceleration of calculations.

Let us summarize the specificity of the proposed approach for computing an urban DEM. *The preliminary aligned stereo pair is used whereas the epipolar geometry is not. The matching procedure is carried out by minimizing the sums of brightness absolute differences over square windows.*

In such a form, the best matching optical flow algorithm turns out to be programmable via MMX or SSE2 instructions.

Since standard digital gray scale images are represented as byte arrays, arithmetical operation, including computation of the proposed version of the optical flow, can be performed using MMX or SSE2 instructions. In this case the execution time of the algorithm is reduced by several orders of magnitude.

In order to explain the technique in details let us introduce some notations.

Denote by $S = \{0,...,n_1-1\}\times\{0,...,n_2-1\}$ the common set of pixels $\mathbf{p}=(p_1,p_2)\in S$ of both images $\mathbf{I}(1)$ and $\mathbf{I}(2)$ composing the stereo pair (as we have assumed that images are preliminary aligned, $\mathbf{I}(1)$ and $\mathbf{I}(2)$ are the aligned common parts of the original stereo pair). The pixel brightness is denoted by $I_{\mathbf{p}}(i)\in\{0,...,255\}$, $i=1,2$. The window $W_\ell(\mathbf{p})$ of half-size ℓ centered in pixel $\mathbf{p}=(p_1,p_2)$ is the set of pixels

$$W_\ell(\mathbf{p}) = \{\mathbf{q}=(q_1,q_2)\,|\,\max\{|p_1-q_1|,|p_2-q_2|\}\le\ell\}.$$

The disparity between two pixels $\mathbf{p}_1$ and $\mathbf{p}_2$ is the difference $\mathbf{d}=\mathbf{p}_2-\mathbf{p}_1$. The windowed matching function is given by

$$F_{\mathbf{p}}(\mathbf{d}) = \sum_{\mathbf{q}\in W_\ell(\mathbf{p})}\left|I_{\mathbf{q}}(1)-I_{\mathbf{q}+\mathbf{d}}(2)\right|. \qquad (1)$$

Its values generate the matrix $\mathbf{F}(\mathbf{d})$. Traditionally, $F_{\mathbf{p}}(\mathbf{d})$ is called the Sum of Absolute Difference (SAD).

Let, for some fixed integer number m, which is half-size of admissible shifts, the value of the optical flow $\hat{\mathbf{d}}(\mathbf{p})$ for pixel $\mathbf{p}$ be

$$\hat{\mathbf{d}}(\mathbf{p}) = \hat{\mathbf{d}}_m(\mathbf{p}) = \underset{|\mathbf{d}|\le m}{\arg\min}\{F_{\mathbf{p}}(\mathbf{d})\},\quad \mathbf{p}\in\tilde{S}$$

where the minimization is performed on the set of pixels

$$\tilde{S} = \{\mathbf{p}\,|\,\ell+m\le p_1<n_1-\ell-m,\quad \ell+m\le p_2<n_2-\ell-m\}.$$

Then, the dense disparity map can be presented as the matrix $\mathbf{D}=\{\hat{\mathbf{d}}(\mathbf{p})\}_{\mathbf{p}\in\tilde{S}}$.

To directly compute $\mathbf{D}$, the number of absolute values of differences in (1), which in the sequel are called operations, is equal to

$$N_{\mathbf{D}} = (2m+1)^2(2\ell+1)^2\tilde{n}_1\tilde{n}_2,$$

where $\tilde{n}_i = n_i - 2(\ell+m), i=1,2$.

Several approaches have been proposed to decrease $N_{\mathbf{D}}$ for estimating the dense disparity map $\mathbf{D}$. Most of them consist in finding approximate solutions, but a few address exact computations [14]. One of them is the ***cutoff*** method [14, 15]. Briefly, it can be formulated in the following form. During computation of the disparity in some pixel $\mathbf{p}$ at iteration $t, \left(t\le(2\ell+1)^2\right)$ we store the minimum value $F_{\mathbf{p}}(\mathbf{d}_{\tau(\min)})$ of $F_{\mathbf{p}}(\mathbf{d}_\tau)$ over all already tested displacement $\mathbf{d}_\tau, \tau<t$. The computation of the current sum (1) is stopped if its value exceeds $F_{\mathbf{p}}(\mathbf{d}_{\tau(\min)})$.

If the current sum $F_{\mathbf{p}}(\mathbf{d}_t)$ is lower than the previously computed minimum value, it is fixed as the current minimum. Such a simple technique for computing large SADs $F_{\mathbf{p}}(\mathbf{d})$ greatly accelerates the estimation of a dense disparity map, especially for large search windows. The method we use as a reference for comparison consists in estimating the dense disparity map by a direct computation of the SAD, using the ***cutoff*** technique. It is referred to as ***Strategy 1***.

We compare this basic approach to different strategies. ***Strategy 2*** consists in computing the sum of absolute values $F_{\mathbf{p}}(\mathbf{d})$ by SSE2 instructions, but without cutoffs [15]. ***Strategy 3*** considers SSE2 instructions for the SAD and the ***cutoff*** technique to estimate $\mathbf{F}(\mathbf{d})$.

We also consider a more sophisticated strategy, which consists in keeping in memory the values of $\left|I_{\mathbf{q}}(1)-I_{\mathbf{q+d}}(2)\right|, \mathbf{q}\in\tilde{S}$ for each fixed shift vector $\mathbf{d}=(d_1,d_2)$ when computing the entire SAD matrix $\mathbf{F}(\mathbf{d})$. This allows decreasing the number of operation by a factor of $(2\ell+1)^2$. For example, for windows $W_\ell(\mathbf{p})$ of half size $\ell=10$ and $\ell=25$, the number of operations decreases by 441 and 2601 times respectively. The number of operations is even more decreased by using the ***running sum*** method [16] for determining $\mathbf{F}(\mathbf{d})$.

The running sum technique was introduced to avoid repeating summation of the same $\left|I_{\mathbf{q}}(1)-I_{\mathbf{q+d}}(2)\right|, \mathbf{q}\in\tilde{S}$ for different $F_{\mathbf{p}}(\mathbf{d})$ [8]. Consider two neighbors SAD $F_{(p_1,p_2)}(\mathbf{d})$ and $F_{(p_1+1,p_2)}(\mathbf{d})$. The window $W_\ell(p_1+1,p_2)$ differs from $W_\ell(p_1,p_2)$ in two columns. Therefore, to get $F_{(p_1+1,p_2)}(\mathbf{d})$ we can subtract from $F_{(p_1,p_2)}(\mathbf{d})$ summands its left column, and add summands of the right column of $F_{(p_1+1,p_2)}(\mathbf{d})$. Actually, it requires only $2(2\ell+1)$ operations instead of $(2\ell+1)^2-1$ for direct summation. If we, first, use the ***running sum*** method to summarize elements of the left and the right columns we have approximately 4 operations for each window.

That is our ***Strategy 4*** relying on storing all $\left|I_{\mathbf{q}}(1)-I_{\mathbf{q+d}}(2)\right|, \mathbf{q}\in\tilde{S}$ for the current $\mathbf{d}$ and using the ***running sum*** method. It is implemented without MMX or SSE commands.

For any of these optimization strategies the number of operations is huge. However, the optical flow computation can be even more accelerated by using techniques which allow the SAD computation without conditional operators. This is the case when using MMX and SSE2 instructions in the concurrent mode [15]. For example, the computations of absolute values of differences for two images of size 640x480 takes only 0.086 milliseconds (ms) when using SSE2 on Intel Core 2 Duo T7500 2.2 GHz.

Strategy 5 is a first modification of ***Strategy 4.*** We propose to use MMX or SSE2 instructions only in the step consisting in computing the SAD but to implement the ***running sum*** *methods* [8] without these instructions.

Finally, ***Strategy 6*** consists in using MMX or SSE2 instructions for both computing the SAD and implementing the ***running sum*** method. This strategy turns out to be the fastest, as shown on Table 1. For its implementation all summands $\left|I_{\mathbf{q}}(1)-I_{\mathbf{q+d}}(2)\right|, \mathbf{q}\in\tilde{S}$ for each fixed shift $\mathbf{d}$ are computed by SSE2 commands and stored. Then the matrix $\mathbf{G}(\mathbf{d})=\{G_{\mathbf{p}}(\mathbf{d})\}_{\mathbf{p}\in\tilde{S}}$ with entries

$$G_{\mathbf{p}}(\mathbf{d})=\sum_{k=-\ell}^{\ell}\left|I_{(p_1,p_2+k)}(1)-I_{(p_1+d_1,p_2+d_2+k)}(2)\right| \qquad (2)$$

is calculated by the SSE2 version of the ***running sum*** method. Summation (2) takes no more than $2n_1n_2$ operations which are performed in concurrent mode.

To compute the matrix $\mathbf{F}(\mathbf{d})$ with entries $F_{\mathbf{p}}(\mathbf{d})$, the MMX or SSE2 realization of the running sum method, can be applied once more to the matrix $\mathbf{G}(\mathbf{d})$. However, in the current version it has not been realized yet. Again, it takes no more than $2n_1n_2$ operations if performed in concurrent mode. So, the total number of operations is less than $4n_1n_2$. It is interesting to note that the number of operations does not depend on the size of the window used for the SAD. This fact was confirmed practically on computation time. The computation times, associated with each strategy, are summarized in Table 1.The flow chart of ***Strategy 6*** is depicted in Fig. 3.

A similar technique is proposed in [17, 18]. Our approach differs from it by using the SAD instead of the windowed correlation, which requires slower operations such as multiplication, division and the square root. Moreover, the L^1 norm is better suited for urban landscapes, which contain numerous edges.

for all $\mathbf{d}\in\mathring{W}_m$

for all $\mathbf{p}\in\tilde{S}$

begin

$\Delta(\mathbf{d})\leftarrow$ *SSE_Diff*$(\mathbf{d},\mathbf{I}(1),\mathbf{I}(2))$;

$\mathbf{G}(\mathbf{d})\leftarrow$ *SSE_Run_Col_Sum*$(\Delta(\mathbf{d}))$;

$\mathbf{F}(\mathbf{d})\leftarrow$ *Run_Wind_Sum*$(\mathbf{G}(\mathbf{d}))$;

$\hat{\mathbf{D}}(t)=\arg\min_{\mathbf{d}(\tau),\,\tau<t}\{F_{\mathbf{p}}(\mathbf{d})\}_{\mathbf{p}\in\tilde{S}}$

end

STOP

Fig.3 Flow chart of *Strategy 6.*

The function *SSE_Diff*($\mathbf{d}$,$\mathbf{I}$(1),$\mathbf{I}$(2)) computes the SAD matrix $\Delta(\mathbf{d})$ of the aligned stereopair $\mathbf{I}(1)$,$\mathbf{I}(2)$, the function *SSE_Run_Col_Sum*($\Delta(\mathbf{d})$) finds sums of columns $G_{\mathbf{p}}(\mathbf{d})$ whereas *Run_Wind_Sum*($\mathbf{G}(\mathbf{d})$) determines the matrix $\mathbf{F}(\mathbf{d})$.

The matrix

$$\hat{\mathbf{D}}(t)=\arg\min_{\mathbf{d}(\tau),\,\tau<t}\{F_{\mathbf{p}}(\mathbf{d})\}_{\mathbf{p}\in\tilde{S}}$$

is the matrix of the region-based matching at current time t for shifts $\mathbf{d}(\tau)$ that are looked through before t.

Table 1. Execution time of different strategies computed by one core of PC Intel Core 2 Duo T7500 2.2 GHz

Image size	**Shift Size**	**Window Size**	**Complexity**	**Strategies**					
				Direct with cutoff	Direct with SSE2 for SAD	Direct with SSE2 for SAD and cutoff	Storing SAD and running sum	SSE2 for SAD, storing SAD, running sum	SSE2 for SAD, storing SAD, SSE2 for running sum
				1	2	3	4	5	6
1365 × 948	13 × 13	9×9	N	0:00:37	–	–	0:00:06	0:00:05	0:00:03
		15×15	~ 2.8 N	0:01:30	0:00:30	0:00:25	0:00:06	0:00:05	0:00:03
		33×33	~ 13 N	0:05:47	0:01:40	0:01:31	0:00:06	0:00:05	0:00:03
2730 × 1896	25 × 25	17×17	~ 53 N	0:24:11	0:06:18	0:05:04	0:01:18	0:01:14	0:00:46
5460 × 3792	49 × 49	17×17	~ 203 N	5:53:11	2:28:38	2:03:52	0:23:15	0:22:32	0:12:16
		33×33	~ 764 N	21:22:31	6:22:00	5:47:28	0:23:15	0:22:32	0:12:16

All presented strategies were tested to compute the urban DEM shown on figure 4.

3. COMPUTING AN URBAN DEM

As we mentioned above our goal was an optimization of computation time for estimating a dense disparity map associated to an urban DEM.

To simplify computations a preliminary alignment of stereo pair images, shown in Fig.1,2, using a simple shift has been performed.

Disparities of the aligned images are estimated not to exceed 5-6 pixels. This property allows using the optical flow technique without considering the epipolar geometry.

The region-based matching version of the optical flow was applied for its simplicity, which leads to significant acceleration.

The computation strategies of the region-based matching mentioned above were used with square windows of sizes 3÷33.

The optimization technique seriously shortened time of determining disparities. It made possible application of subpixel image resolution. In this context, we increased the stereo image size by factors 2, 3, 4 and 5.

When comparing the nearest neighbor, bilinear and bicubic interpolations to increase the image size, we concluded that using the nearest neighbor interpolation provides the best results. This behavior is probably due to the fact that the nearest neighbor interpolation smoothes less image boundaries, which are an essential feature in urban scenes.

Computation times are described in Tables 1,2.

Table 2. Parallel Execution on Strategy 6 by many cores

Image size	**Shift Size**	**Window Size**	**Intel Core 2 Quad Q6600 @ 2.4 GHz**			
			1x	**2x**	**3x**	**4x**
2730×1896	25×25	9×9	0:01:16	0:00:42	0:00:31	0:00:24

The first Table shows execution time of the different strategies, whereas the second one maps the dependence of execution time of the parallel versions for the fastest (i.e. ***Strategy 6***) with respect to the number of cores on a multiprocessor Intel Core 2 Quad Q6600 2.4 GHz computer.

Elevation maps for the calibrated stereo pair, which are shown in Fig 1,2, are depicted in Fig.3,4.

4. CONCLUSION

We investigated several modern computational techniques to accelerate the estimation of an urban DEM. The fastest of them, which consists in storing in memory all absolute values of differences of shifted images for each current admissible shift, as well as using the MMX or SSE2 realization of the ***running sum*** method, speeds up drastically the dense disparity map computation as compared with the direct computation (***Strategy 1***).

We plane to take benefit of this low computation time to consider more sophisticated models for the 3D reconstruction. Including prior information, in the optimized functional, compatible with this modern computing technology is currently under study.

5. ACKNOWLEDGMENT

The authors would like to thank to the French institute (IGN) for kindly providing the calibrated stereo pairs of the urban scene. This work was partially funded by INRIA and EGIDE through respectively the associated team ODESSA and the ECONET project 18902PK.

6. REFERENCES

[1] P. Burrough. *Principles of Geographical Information Systems for Land Resources Assessment.* Clarendon Press. Oxford, United Kingdom, 1986. p. 193.

[2] J. Wilson and J. Gallant. *Terrain Analysis: Principles and Applications.* John Wiley and Sons. New York, 2000. p. 512.

[3] F. Lafarge, X. Descombes, J. Zerubia and M. Pierrot-Deseilligny. An automatic 3D city model : a Bayesian approach using satellite images. *I In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (ICASSP`2006), Toulouse, France, 2006.

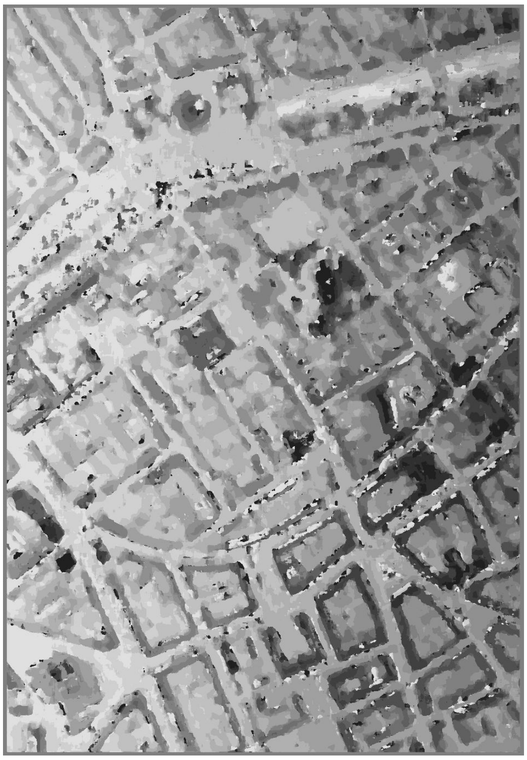

Fig.3 – Calculated disparity map of urban scene

Fig.4 – Calculated DEM of urban scene

[4] F. Lafarge, X. Descombes, Zerubia and M. Pierrot-Deseilligny. Automatic Building Extraction from DEMs using an Object Approach and Application to the 3D-city Modeling. *Journal of Photogrammetry and Remote Sensing*, 63(3) (2008). p. 365-381.

[5] C. Baillard. A stereo matching algorithm for urban digital elevation models, *Photogrammetric Engineering and Remote Sensing* 66(9) (2000). p. 1119-1128.

[6] L. Alvarez, R. Deriche, J. Weickert and X. Sanchez. Dense disparity map estimation respecting image discontinuities: a pde and scale-space based approach, *Journal of Visual Communication and Image Representation* 13(1-2) (2002). p. 3-21.

[7] N. Slesareva, A. Bruhn and J. Weickert. Optic Flow Goes Stereo: A Variational Method for Estimating Discontinuity-Preserving Dense Disparity Maps, *Proceedings of 15th Scandinavian Conference on Image Analysis (SCIA'2007),* Aalborg, Denmark June 2007, pp. 173-182.

[8] C. Sun. Fast optical flow using 3d shortest path techniques, *Image and vision computing* 20(13/14) (2002). p. 981-991.

[9] J. Barron, D. Fleet and S. Beauchemin. Performance of optical flow techniques, *International Journal of Computer Vision* 12(1) (1994). p. 43-77.

[10] P. Zhao and M. Spetsakis. Near Real-Time Optical Flow. *Proceedings of the 14th International Conference on Vision Interface,* Ottawa, Canada 2001, pp. 47-55.

[11] B. Horn *Robot Vision*. MIT Press. Cambridge, 1986. p. 509.

[12] D. Forsyth and J. Ponce. Computer Vision: a modern approach. Prentice Hall. New Jersey, 2003. p. 693.

[13] T. Toivonen A New Algorithm for Fast Full Search Block Motion Estimation Based on Number Theoretic Transforms. *Proceedings of the 9th International Workshop on Systems, Signals, and Image Processing,* Manchester, United Kingdom 2002, pp. 90-94.

[14] Y. Chen, Y. Hung and C. Fuh. Fast block matching algorithm based on the winner-update strategy, *IEEE Transactions on Image Processing* 10(8) (2001), p. 1212-1222.

[15] Intel Corp. Using MMX Instructions to Compute the AbsoluteDifference in Motion Estimation. March 1996.

[16] M. McDonnel. Box-filtering techniques, *Computer Graphics and Image Processing* 17 (3) (1981). p. 65-70.

[17] R. Sadykhov and D. Lamovsky. Fast cross correlation algorithm for optical flow estimation, *Computing* 5(3) (2006). p. 112-117.

[18] R. Sadykhov and D. Lamovsky. Fast cross correlation algorithm for optical flow estimation, *Proceedings of the 7th Nordic Signal Processing Symposium (NORSIG'2006),* Reykjavik, Iceland 7-9 June 2006, pp. 322-325.