J Supercomput (2015) 71:1857–1868
DOI 10.1007/s11227-014-1364-x

Parallelization of a method for dense 3D object reconstruction in structured light scanning

Cristina Portalés · Juan M. Orduña · Pedro Morillo

Published online: 28 December 2014

Abstract Optical three-dimensional shape measurement based on structured light has been widely used for 3D measurement in many different applications. Although many different methods for 3D object reconstruction have been proposed last years, surprisingly none of the proposals includes a parallelization study of the tasks executed on computers, preventing these methods from reaching their maximum performance. In this paper, we propose first the computational evaluation of a previously proposed 3D object reconstruction method. Based on that evaluation, we also propose the parallelization of the reconstruction method using the OpenMP API specification for shared-memory parallel programming. The results show that most of the execution time is consumed by the tasks depending on the I/O hardware (camera, projector, hard disk, etc.), in such a way that the tasks performed by the computer should be overlapped as much as possible with the tasks performed by the I/O hardware, for those scenarios where the acquisition of the images should be performed on-line. For those scenarios where all the images are already available, then the inherent parallelism of the application increases, allowing a reduction of the execution time that can reach up to a 82 %. These results validate the proposed parallelization as a valuable implementation for data centers that provide web services for 3D object reconstruction purposes.

Keywords 3D Object reconstruction · Structured light scanning · Parallel computing · OpenMP

C. Portalés
IRTIC-Universidad de Valencia, Valencia, Spain
e-mail: Cristina.Portales@uv.es

J. M. Orduña (✉) · P. Morillo
Departamento de Informática, Universidad de Valencia, Valencia, Spain
e-mail: juan.orduna@uv.es

P. Morillo
e-mail: Pedro.Morillo@uv.es

1 Introduction

Optical three-dimensional shape measurement based on structured light (SL) has been widely used for 3D measurement, machine vision, automated manufacturing, and industrial monitoring [4,6,7,9]. These techniques offer the advantages of non-contact operation, full-field acquisition, high-resolution and dense 3D reconstruction. SL techniques assume a projection of controlled illumination of the scene through one or more projected patterns onto the objects surface, commonly using DLP or LCD video projectors. Due to the differences in the object height, the projected pattern appears distorted when viewed obliquely. Therefore, the fringe pattern is modulated according to the objects 3D height and the angles formed between the illumination and the viewing axes. Projected light patterns have a distinctive structure (appearance), because the projector image pixels are coded in a certain way. Detecting the same code on the pixels from a minimum of two cameras, or from one camera and one projecting device, the correspondence problem for a large number of points can be solved, leading to dense 3D surface acquisition after triangulation, for which the calibration of the camera(s) and/or projector(s) has to be known.

Different types of SL measurement methods have been developed [11,14]. Among these methods, Phase Shifting (PS) acquisition methods have been used extensively [5,13]. Compared to other structured light techniques, PS techniques offer robustness against ambient light and reflections variation, due to the grayscale nature of the projected patterns [11]. Figure 1 shows the general procedure of 3D object reconstruction in PS-based approaches.

In these techniques, a sequence of periodic intensity patterns is projected onto an object or a surface, each of which is offset by a fraction of its period from the previous one, so that the entire period is covered. However, the demodulation of the acquired fringe patterns results in a so-called wrapped phase instead of the desired (unwrapped) phase. This raises the problem known as a *phase unwrapping* procedure (see Fig. 1). Many different methods have been proposed for unwrapping the phase [5,8,13]. Multiple Phase Shifting (MPS) methods use more than one frequency to cope with the uncertainty created in the extracted wrapped phase. Recently, an effective MPS method which provides high accurate reconstruction shapes has been proposed [13]. Once the unwrapped phase map is obtained, a relationship between the encoded phase and the height of the object should be established. This procedure is known as *phase-to-height conversion* (see Fig. 1), and it requires a calibration procedure of the devices. In this sense, many calibration methods have been developed for that purpose [16–18]. One of these methods [17] uses implicit calibration, i.e., instead of explicitly

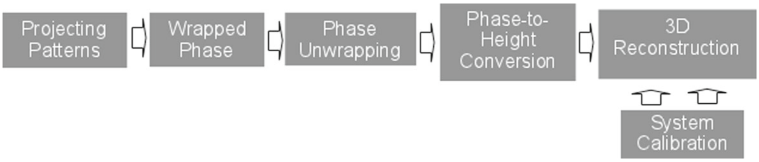

Fig. 1 A general procedure for 3D object reconstruction based on Phase Shifting methods

computing the (interior and exterior) calibration parameters for the devices, a set of coefficients are calculated for a certain system configuration, and a mathematical model describes the relationship between these coefficients and the objects height. In this approach, a 3D calibration object is used, for which their heights in relation to a reference plane are known. The main advantage of this strategy is that the calibration method generates less computational workload than others [16], since the obtained coefficient parameters are independent of the image coordinates.

Regardless the SL method chosen for the reconstructing 3D objects, a common characteristic to all of them is the lack of parallelization studies of the computer-based procedures, in such a way that the overall method performance is not maximized. To best of our knowledge, there are very few works where the parallelization of some procedures is considered as a possible improvement of the reconstruction method, as in [2,15]. The first of these works [2] does not propose code parallelization, but a strategy to improve the efficiency of the sequential code executed on a computer. The second of these works [15] proposes the use of a parallel set of projectors and cameras, but no parallel code is executed on a computer. Others works try to optimize the computational time by providing more efficient methods, but no code parallelization is considered [13,17,19]. Since no computational measurements are provided in the proposals of SL methods, we can only state that they do not provide their best performance, although the improvements that could be achieved through code parallelization are unknown.

In a previous paper [12], we proposed a new method of 3D object reconstruction that combined the benefits shown by Multi Phase Shifting [13] with the potential of the improved algorithm for phase-to-height mapping [17]. In this paper, we propose first the computational evaluation (in terms of execution time) of that reconstruction method, in order to measure the expected performance when executed on standard hardware. Based on this evaluation, we also propose the improvement of that method [12] through its parallelization using the OpenMP API specification for shared-memory parallel programming [1,3], in order to fully exploit the parallelism available in current multicore processors. We propose two different parallelization options, for different scenarios. The results show that for those scenarios where the acquisition of the images should be performed on-line, most of the execution time is consumed by the tasks depending on the I/O hardware (camera, projector, hard disk, etc.), in such a way that the tasks performed by the computer should be overlapped as much as possible with those performed by the I/O hardware. In this way, the parallelization can achieve a reduction up to a 26 % of the execution time with respect to the sequential execution. For those scenarios where all the images are already available, then the inherent parallelism of the application increases, allowing a reduction of the execution time that can reach up to a 60 %.

The rest of the paper is organized as follows: in order to make this paper self-contained, Sect. 2 summarizes the method of 3D object reconstruction [12] whose parallelization is the purpose of this work. Section 3 discusses different options for the parallelization of the method, taking into account that some tasks involve external hardware connected to the multicore computer. Next, Sect. 4 shows the performance evaluation of all the considered options, including the sequential execution of the method as a reference value. Finally, Sect. 5 shows some conclusions and future work to be done.

2 A new method for 3D object reconstruction

In this section, we summarize the proposed method for 3D Object Reconstruction based on Structured light scanning [12], in order to make this paper self-contained. The implemented method of phase unwrapping is based on the one described in [13], which makes use of Multiple Phase Shifts (MPS) patterns. According to their authors, the algorithm is robust to objects with sharp discontinuities and depth changes, and it provides better accuracy in 3D reconstruction than standard Phase Shifting. On the other hand, the method of phase-to-height conversion that we have used is based on the technique described in [17], which provides a phase-to-height mapping with a calibration method that makes use of a 3D reference object. The main advantage of this method is that it uses an implicit procedure, avoiding the need to geometrically calibrate the projector, a massive time consuming task that produces many problems related to the projector lens distortions. Another advantage of this method is that it allows a more flexible system configuration than basic measurement systems in phase measuring profilometry [10,19], since the reference plane does not need to be orthogonal to the camera optical axis and both the camera and the projector can be located at different distances to the reference plane.

Compared to the method described in [13], our proposal adds less computational workload, since the calibration method is much less complex than traditional calibration methods based on the computation of the spatial and geometric properties of the camera and the projector. However, in order to perform MPS, the PS method should be applied twice, and therefore it may generate more computational workload than the one shown in [17]. Concretely, the steps of unwrapping and phase-to-height conversion can be split in three different processes involving different tasks. These tasks are the following ones:

- Process 1: Projection, capture and storage of images.
  Task 1: Projection, capture and storage of images (PCS) with the first period of the periodic intensity pattern for the reference plane.
  Task 2: PCS with the second period for the reference plane.
  Task 4: PCS with the first period for the calibration object.
  Task 6: PCS with the second period for the calibration object.
  Task 9: PCS with the first period for the object to be reconstructed.
  Task 11: PCS with the second period for the object to be reconstructed.
- Process 2: Computing of the wrapped phase.
  Task 3: Computation of the wrapped phase using the PS algorithm (CWP) for the images gathered in Task 1.
  Task 5: CWP for the images gathered in task 2.
  Task 7: CWP for the images gathered in task 4.
  Task 10: CWP for the images gathered in task 6.
  Task 12: CWP for the images gathered in task 9.
  Task 14: CWP for the images gathered in task 11.
- Process 3: Computing of the object height.
  Task 8: Computation of the absolute unwrapped phase using the MPS algorithm (CUP) for the phases computed in tasks 3 and 5.

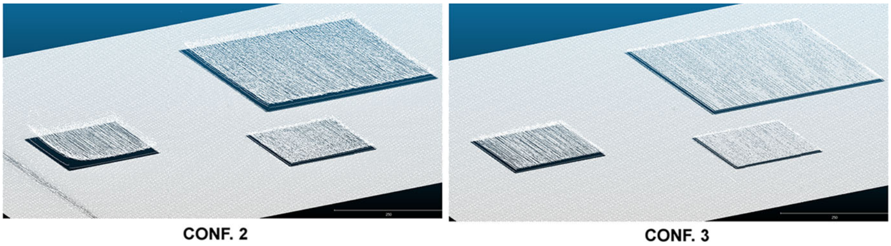

Fig. 2 Visualization of the 3D cloud of points achieved by each of the three configurations

Task 13: CUP for the phases computed in tasks 7 and 10.
Task 15: Computation of the seven calibration parameters from the unwrapped phases computed in tasks 8 and 13.
Task 16: CUP for the phases computed in tasks 12 and 14.
Task 17: Computation of the object height with the parameters computed in task 15 and the unwrapped phases computed in tasks 8 and 16.
Task 18: Computation of the cloud of points from the object height computed in task 17.

In order to design the most efficient method, we considered three configurations [12]. We compared different options in our proposal, in order to select the best combination of elements in the 3D object reconstruction method. Concretely, we considered different patterns to be projected, and different phase unwrapping techniques for each configuration. The configuration that provided the best results was the one using PS technique [13] for achieving a wrapped phase, as well as the phase-to-height method proposed in [17]. For the phase unwrapping step, the best option was MPS [13]. For illustrative purposes, Fig. 2 depicts the 3D reconstructed cloud of points for two of the considered configurations. This figure shows that in Configuration 2 (another configuration considered for comparison purposes) erroneous geometries are produced at the edges of the boxes, like in most of the PS-based methods. However, Configuration 3 (the one proposed as the new method for 3D object reconstruction) can deal with the sharp edges of the 3D reconstructed object, removing the erroneous geometries.

3 Parallelization

In this section, we discuss the possible options in order to fully exploit the parallelism available in current multicore processors. The first step is to measure the performance obtained by the sequential execution of the application. Then, we present different parallelization strategies, based on OpenMP, in order to take advantage of the multicore processors currently available in standard computer platforms.

3.1 Sequential evaluation

We have evaluated the performance of the proposed algorithm (in terms of execution times) for different computer platforms. We have measured the execution times for a

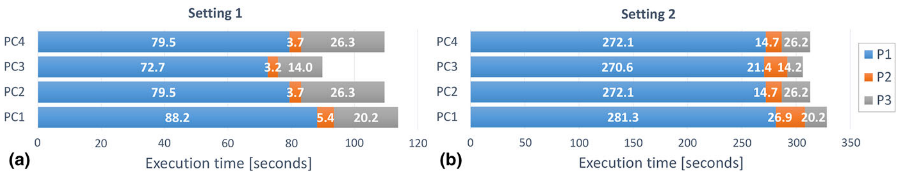

Fig. 3 Execution times for the settings 1 (**a**) and 2 (**b**)

sequential implementation of the 18 tasks described above using Configuration 3, when executed on four different standard computer platforms: the first platform is based on an Intel Core(TM)2 4400 processor running at 2.00 GHz, and with 4 GB of RAM. The second platform is based on an AMD Athlon II X3 445 Processor running at 3.10 GHz, and 4 GB of RAM. The third platform is based on an Intel Core Quad Q9550 running at 2.83 GHz, and 4 GB of RAM. Finally, the fourth platform is based on an Intel i7-3930K processor, with 6 cores and 12 threads and 48 GB of RAM. The four platforms work on Windows 7 64 bits. The acquisition hardware was composed of one Logitech webcam (max. resolution of 1,600 × 1,200 pixels), and a Vivitek DLP projector (max. resolution of 1,800 × 1,200 pixels). Although the sequential implementation executes the three processes in an orderly manner (first P1, then P2 and then P3), in order to study the relative workload that these processes add to the application we have measured not only the overall execution time, but also the execution time required by each process. For each platform considered, we have evaluated the proposed method with four different settings:

Setting 1: Offset for T32 of 4 pixels (8 projected images); offset for T45 of 5 pixels (9 projected images); gathered images of 800 × 600 pixel resolution.
Setting 2: Offset for T32 of 1 pixel (32 projected images); offset for T45 of 1 pixel (45 projected images); gathered images of 800 × 600 pixel resolution.
Setting 3: Offset for T32 of 4 pixels (8 projected images); offset for T45 of 5 pixels (9 projected images); gathered images of 1,600 × 1,200 pixel resolution.
Setting 4: Offset for T32 of 1 pixel (32 projected images); offset for T45 of 1 pixel (45 projected images); gathered images of 1,600 × 1,200 pixel resolution.

Settings 1 and 3 are the typical settings used in many other proposals, which assume high- and low-resolution images (obtaining point clouds with a higher or lower density), respectively, and a low number of projected images. Additionally, we have included setting 2 and setting 4, in order to study the worst case (the projection of all the possible images, with the purpose of obtaining the best optical quality). In this way, we guarantee that the execution times shown below are the longest ones that are obtained by the computer platforms for the considered configurations.

Figure 3a shows the results obtained for setting 1. This figure shows on the X axis the execution times required by the proposed method. On the Y-axis, this figure shows the different computer platforms considered for evaluation purposes. For each platform, there is a single bar, composed of three segments, one for each of the processes considered (the leftmost segment corresponds to Process 1, the central segment corresponds to Process 2, and the rightmost segment corresponds to Process 3). Each segment displays a number corresponding to the execution time of that Process (in

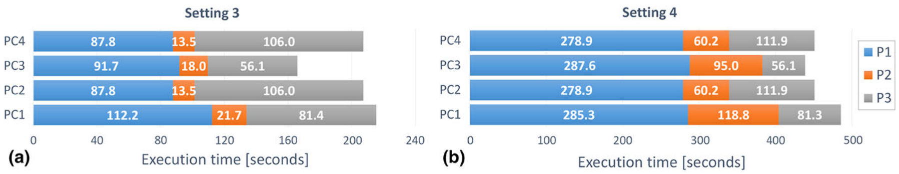

Fig. 4 Execution times for settings 3 (**a**) and 4 (**b**)

seconds). Thus, the sum of the lengths of the three segments is the total length (execution time) of the 3D reconstruction method, since we have performed a sequential execution of the processes. This Figure shows that the relative execution time of the processes is unevenly distributed. Process 1 requires most of the execution time, being more than twice of the time required by Process 3 and ten times the one required for Process 2. Also, Fig. 3a shows that there are significant differences among the computer platforms, providing overall execution times varying from 115 to 90 s (around 22 % of reduction with respect to the longest time provided by PC1). Surprisingly, the most powerful computer platform does not provide the shortest execution times. In the case of setting 2, Fig. 3b shows that, as it could be expected, the time for acquiring 32 projected images instead of 8 images greatly increases the execution time of Process 1 (and the overall time). As a result, Process 1 takes most of the execution time, making useless any parallelization of processes 2 and 3 if this low-cost acquisition hardware is deployed. Nevertheless, the deployment of faster devices may change this scenario.

Figure 4 shows the results obtained for settings 3 and 4. This figure shows that for these settings there are also significant differences among the computer platforms, being the PC3 the platform that provides the shortest execution times for both settings, and PC1 the one providing the longest ones. When comparing the results for setting 1 and 3, it can be seen that the overall execution times shown in Fig. 4a are around twice the ones in Fig. 3a. It is also worth mention that for images with higher resolutions (Fig. 4), the computational workload supported by the computer platforms becomes much higher, in such a way that the relative fraction of the overall execution time consumed by processes 2 and 3 in Fig. 4a reaches two thirds of the overall time (in the case of PC2). These results suggest that for this setting any parallel implementation of processes 1, 2, and 3 would significantly improve the performance in regard to the sequential implementation. In the case of Fig. 4b, the relative fraction of the overall execution time consumed by processes 2 and 3 also increases with respect to the ones in setting 2 (Fig. 3b), but the number of images to be processed prevents this relative fraction to exceed 42 % of the overall time.

Given the similarity among the results obtained by the four considered platforms (PC1–PC4), in the rest of the paper we will show the results only for the most up-to-date platforms, PC3 and PC4.

3.2 Parallel strategies

The results shown in Sect. 3.1 show that the external hardware and the sequential tasks are the system bottleneck, requiring more than half of the execution time for any of

Table 1 Execution of tasks in strategy 1

Thread 1	1	2	4	6	9	11				
Thread 2		3	5, 8	7	10, 13, 15	12	14	16	17	18

Table 2 Execution of tasks in strategy 2

Offline	1, 2, 4, 6, 9, 11					
Thread 1		3	8	15	17	18
Thread 2		5	13	16		
Thread 3		7				
Thread 4		10				
Thread 5		12				
Thread 6		14				

the settings. These results suggest that the parallelization should overlap as much as possible the tasks in processes 2 and 3 with the tasks in Process 1. This will be the case for those situation where the 3D reconstruction method should be applied ad-hoc. Nevertheless, there are some other situations where the tasks in Process 1 can be performed "off-line." For example, data centers or cloud computing facilities where the projected, captured and stored images arrive, and the reconstructed 3D images are expected to be delivered as soon as possible. Therefore, we have decided two different parallelization strategies, one for each case. Thus, in Strategy 1, where the tasks in Process 1 should be performed on-line, all the tasks in processes 2 and 3 are scheduled as soon as the data dependencies allow, in order to overlap these tasks as much as possible with the ones in Process 1. Nevertheless, the data dependencies among tasks in Strategy 1 (especially the sequential execution of all the tasks in Process 1) result in a parallelization consisting of only two threads, one that executes all the tasks in Process 1 and the other one for executing the tasks in processes 2 and 3. For illustration purposes, Table 1 shows a schematic view of Strategy 1, and how the tasks are programmed as two parallel threads. This view assumes that time passes from left to right, and the number of parallel threads is represented by the existing number of rows in the table, in such a way that a sequential implementation would consist of a single row with a length of 18 tasks. This table shows that the inherent degree of parallelism shown by the method is not significant. On the one hand, all the tasks in process 1 should be sequentially executed, since they depend on the external hardware. On the other hand, two or three tasks in processes 2 and 3 can be grouped and executed as a single task (tasks 5 and 8, and tasks 10,13 and 15), at two different points of the procedure, due to dependencies among them and in regard to tasks in Process 1. Due to these constraints, the main advantage of this strategy is expected to come from hiding the latency of Process 1, but it can take advantage of only two of the cores in the multicore processor (it is only split into two parallel threads).

In Strategy 2, where the results of all the tasks in Process 1 are assumed to be available at the same time, the six tasks in processes 2 and 3 with an exclusive dependence on Process 1 can be executed in parallel, improving the parallelism of the procedure. Table 2 shows a schematic view of this strategy, and how the tasks are programmed.

Table 3 Execution times required for strategy 1

Platform	Setting 1			Setting 2		
	Time	Reduc. (%)	Ideal (%)	Time	Reduc. (%)	Ideal (%)
PC3	76.23	15.2	19.2	275.91	10.1	11.6
PC4	83.04	24.2	27.4	280.786	10.3	13.1
	Setting 3			Setting 4		
	Time	Reduc. (%)	Ideal (%)	Time	Reduc. (%)	Ideal (%)
PC3	126.042	23.9	44.7	323.547	26.3	34.5
PC4	110.507	46.7	48.9	320.348	29.0	38.2

In this case, up to six tasks can be executed in parallel at a given point of the procedure (they can be programmed as six parallel threads, taking advantage of the existence of up to six cores in the processor), suggesting that in this case the parallelization of the 3D reconstruction method can provide a more significant performance improvement. Both parallelization strategies have been implemented using the "sections" and "section" directives of the OpenMP specification.

4 Performance evaluation

In this section, we show the performance evaluation of the two parallelization strategies described in the previous section. In order to achieve this goal, we have executed the parallel implementations based on the OpenMP specification on the same computer platforms and with the same settings used in the sequential evaluation, for comparison purposes.

Table 3 shows the results provided by the parallelization Strategy 1 for the considered settings. This table shows seven columns. The most-left column shows the platform used for the evaluation (PC3 or PC4). Next, a group of three columns each shows the results for each of the considered settings. For each of the settings, the table shows three columns. The column labeled as "Time" shows the execution time required for that setting when executing the tasks (arranged as indicated in strategy 1) on the corresponding platform. The column labeled as "Reduc (%)" shows the percentage of reduction that this execution time represents in regard to the execution times shown in Figs. 3 and 4 (that is, the percentage of reduction achieved with respect to the sequential execution of the tasks). Finally, the column labeled as "Ideal (%)" shows the ideal percentage of reduction that could be achieved if if all possible overlaps are done and they not add overhead. Since Strategy 1 considers the tasks involving external hardware, in this case the ideal case would be when the execution time is reduced to the execution time of P1.

Table 3 shows that the percentage of reduction ranges from 53.5 % of the ideal percentage of reduction for Setting 3, PC3 (where a 23.9 % is achieved, out of an ideal 44.7 %), to a 95.5 % of the ideal percentage of reduction for Setting 3, PC4 (where a

Table 4 Execution times required for strategy 2

Platform	Setting 1			Setting 2		
	Time	Reduc. (%)	Ideal (%)	Time	Reduc. (%)	Ideal (%)
PC3	11.578	32.8	94.2	15.203	57.2	97.3
PC4	6.412	78.6	91.2	7.691	81.2	96.5
	Setting 3			Setting 4		
	Time	Reduc. (%)	Ideal (%)	Time	Reduc. (%)	Ideal (%)
PC3	44.094	40.5	86.6	60.656	59.9	92.7
PC4	25.084	79.0	81.6	29.989	82.6	89.5

46.7 % is achieved, out of an ideal 48.9 %). These results show the importance of the underlying platform in the achievement of an efficient parallelization. Nevertheless, the problem with this strategy, is that the ideal percentage of reduction is constrained by the sequential use of external hardware. Thus, the maximum ideal reduction percentage is limited to 48.8 % of the sequential execution time, for the case of PC4 and Setting 3. Also, it is worth mention that for low resolution images (Settings 1 and 2), in this strategy the percentage of reduction in the execution time is limited to 24.2 % for setting 1 (PC4), and around 10.3 % for setting 2. Therefore, we can conclude that the benefits of parallelization when using strategy 1 can only reach significant values if high resolution images are used. Otherwise, the limits imposed by the sequential use of external hardware severely limits the potential of parallel implementations.

Table 4 shows the results provided by the parallelization strategy 2 for the considered Settings and platforms. In strategy 2, the tasks in Process 1 are assumed to be performed off-line (they are performed in advance and all their results are available). Therefore, we have computed the ideal percentage of reduction in the execution time as the ratio of the time required for sequentially executing tasks 3, 8, 15, 17, and 18 (as shown in Table 2) with respect to the sequential execution time shown in Figs. 3 and 4.

Table 4 shows that the differences in the performance shown by the different platforms increase in regard to Strategy 1, since in this case the bottleneck of external hardware has been removed. Thus, the percentage of reduction corresponding to platform PC3 that come closest to the ideal percentage of reduction only reaches a 64.6 % of that value (the one for Setting 4, reaching a 59.9 % out of an ideal percentage of 92.7 %). However, platform PC4 reaches a 96.8 % of the ideal percentage for Setting 3 (a 79.0 % out of an ideal percentage of 81.6 %), and at least a 84.1 %, for the case of Setting 2.

Finally, it must be taken into account that the cost of field works in metrology highly depend on the duration of the scanning process. Any reduction in the time required for the scanning process can save significant costs, and therefore the improvements achieved in Strategy 1 are significant, although much lower than the ones achieved in Strategy 2. These reductions validate the proposed parallelization as a very efficient alternative, providing very substantial performance improvements in the 3D object reconstruction method.

5 Conclusions and future work

In this paper, we have proposed the computational evaluation and the parallelization of a previously proposed 3D object reconstruction method using the OpenMP API specification. The results show that most of the execution time is consumed by the tasks depending on the I/O hardware (camera, projector, etc.), in such a way that the tasks performed by the computer should be overlapped as much as possible with those performed by the I/O hardware, for those scenarios where low-cost acquisition hardware is deployed. In these cases, the parallelization can achieve a reduction up to a 46 % of the execution time with respect to the sequential execution. For those scenarios where all the images are already available, then the inherent parallelism of the application increases, allowing a reduction of the execution time that can reach up to a 82 %. These results validate the proposed parallelization as a valuable implementation for data centers that provide 3D object reconstruction web services, and for online systems with significantly faster I/O devices.

Since the high level parallelization is limited by the sequential nature of Process 1, as a future work we plan to study a low level parallelization of each task involved in the procedure.

References

1. Chapman B, Jost G, van der Pas R (2007) Using OpenMP: portable shared memory parallel programming (scientific and engineering computation). The MIT Press, Cambridge
2. Chen S, Li Y, Zhang J (2008) Vision processing for realtime 3-d data acquisition based on coded structured light. Image Process IEEE Trans 17(2):167–176
3. Dagum L, Menon R (1998) OpenMP: an industry standard API for shared-memory programming. IEEE Comput Sci Eng 5(1):46–55
4. de Andrade D, Nakamura R (2013) A marker-free calibration and registration process for multiple depth maps from structured light sensors and its application in video avatar systems. In: XV Symposium on virtual and augmented reality, IEEE, pp 73–82
5. Fu Y, Luo Q (2011) Fringe projection profilometry based on a novel phase shift method. Opt Express 19(22):21,739–21,747. doi:10.1364/OE.19.021739
6. Fu Y, Wang Y, Wan M, Wang W (2013) Three-dimensional profile measurement of the blade based on surface structured light. Optik-Int J Light Electron Opt 124(18):3225–3229
7. Gorthi SS, Rastogi P (2010) Fringe projection techniques: whither we are? Opt Lasers Eng 48(2):133–140 Fringe projection techniques
8. Judge T, Bryanston-Cross P (1994) A review of phase unwrapping techniques in fringe analysis. Opt Lasers Eng 21(4):199–239
9. Maldonado TJ, Lasenby J (2011) Simulation of breathing for medical applications. SIGGRAPH Posters, ACM, p 7
10. Meng S, Ma H, Zhang Z, Guo T, Zhang S, Hu X (2011) Complete calibration of a phase-based 3d imaging system based on fringe projection technique. In: Proceedings of international conference on optical instruments and technology: optoelectronic imaging and processing technology, pp 82,000L–82,000L–7
11. Navarro SF (2009) 3d reconstruction of object shape using structured light. Ph.D. thesis, Department of Computer Architecture and Technology, University of Girona, Spain
12. Portalés C, Morillo P, Orduña JM (2014) Towards an improved method of dense 3d object reconstruction in structured light scanning. In: Proceedings of international conference on computational and mathematical methods in science and engineering (CMMSE), pp 992–1001
13. Pribanic T, Mrvos S, Salvi J (2010) Efficient multiple phase shift patterns for dense 3d acquisition in structured light scanning. Image Vis Comput 28(8):1255–1266

14. Salvi J, Fernandez S, Pribanic T, Llado X (2010) A state of the art in structured light patterns for surface profilometry. Pattern Recognit 43(8):2666–2680
15. Weinmann M, Schwartz C, Ruiters R, Klein R (2011) A multi-camera, multi-projector super-resolution framework for structured light. In: 3D imaging, modeling, processing, visualization and transmission (3DIMPVT), 2011 international conference on, pp 397–404
16. Wen Y, Li S, Cheng H, Su X, Zhang Q (2010) Universal calculation formula and calibration method in fourier transform profilometry. Appl Opt 49(34):6563–6569
17. Xiao Y, Cao Y, Wu Y (2012) Improved algorithm for phase-to-height mapping in phase measuring profilometry. Appl Opt 51(8):1149–1155
18. Zhang B, Li YF, Wu YH (2007) Self-recalibration of a structured light system via plane-based homography. Pattern Recognit 40(4):1368–1377
19. Zhang Z, Ma H, Zhang S, Guo T, Towers CE, Towers DP (2011) Simple calibration of a phase-based 3d imaging system based on uneven fringe projection. Opt Lett 36(5):627–629. doi:10.1364/OL.36.000627