A PARALLEL STRATEGY TO IMPROVE THE CALCULATION OF FORM FACTORS

Gladimir Guimarães Baranoski and Leonardo Garcia
Instituto de Informática - CPGCC - UFRGS
Caixa Postal 15064 - Porto Alegre - RS - Brasil
gladimir@inf.ufrgs.br *garcia@inf.ufrgs.br*

ABSTRACT

This paper presents a parallel strategy to improve the calculation of form factors. It consists of the mixed use of the Parametric Differential Method and the Cooperative Processes Software. An application of this strategy is described and the results are discussed.

CR Categories and Subject Descriptors: D.1.3 [**Concurrent Programming**]: Parallel Programming; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism.

Key Words: form factor, radiosisty and parallelism.

INTRODUCTION

In recent years radiosity methods have become a reference for high-quality rendering. The calculation of form factors represents the bottleneck of these methods, being directly associated with the accuracy and the processing time.

New algorithms have emerged in order to improve the accuracy of the form factors [Baum et al.89], [Wallace et al.89]. In addition, several parallel solutions have already been proposed to provide time savings [Chalmers-Paddon91], [Drucker-Schroder92], [Feda-Purgathofer91]. The current parallel approaches focus on well suited architectures for parallelism. They are at most based on the calculation of form factors by the hemicube method. Unlike these approaches this paper proposes a parallel strategy based on software rather than hardware and on the use of a more accurate method for the calculation of form factors.

The method used on the determination of the form factors is the *Parametric Differential Method (PDM)* [Baranoski92]. The form factors obtained through this method present a high degree of accuracy. If the surfaces of an environment are divided into a finer grid of sectors, named patches, the PDM improves the accuracy of the form factors. The problem lies in the fact that the corresponding processing time increases at a rate higher than the accuracy improvement rate.

The proposed solution to speed-up the calculation of the form factors by the PDM is the use of the *Cooperative Processes Software (CPS)* [Fermilab89], [Nash et al.88]. In a broad sense, it includes a set of facilities which

have been developed to aid the design and the execution of parallel applications. Through the Mixed use of the PDM and the CPS, henceforth referred to as the MPC strategy, we can get, in a reasonable processing time, very accurate form factors regarding an environment divided in a finer grid of patches. The use of CPS on the calculation of form factors has not been discussed in the literature to date and the MPC strategy represents an initial effort in this area.

THE PARAMETRIC DIFFERENTIAL METHOD

The PDM main goal is to allow the application of the radiosity method to the image synthesis of objects formed by curved surfaces without achieving approximations by polygons.

It computes the form factors between surfaces taking into account their parametric descriptions. The surfaces are divided into patches in order to increase the accuracy of the form factors and to provide more accurate radiosities for sectors of the surfaces. Regarding curved surfaces, the division into curved patches, instead of polygons, allows the use of the exact descriptions of the surfaces during the rendering.

The calculation of form factors between finite surfaces involves the resolution of double area integrals. The PDM, like other methods of form factors calculation, solve these integrals using a numerical integration method. The main difference between the PDM and the usual methods lies on the transformation of the mathematical expression of the form factors in order to make it more general. Its terms are expressed in a vector form and take into account the parametric variables used to describe the surfaces. The integrals are evaluated along the limits of the parametric variables without performing approximations of curved contours by line segments.

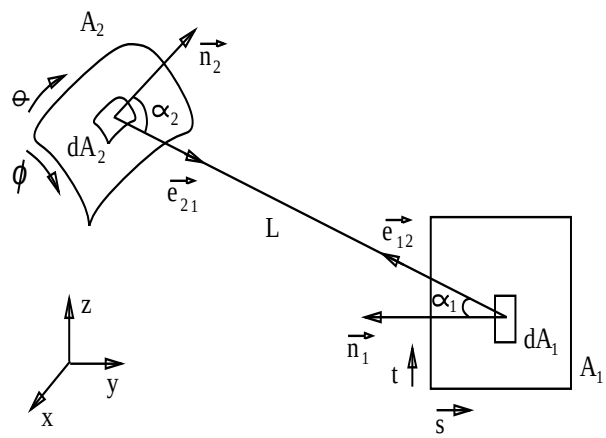
The equation that defines the form factor between two finite parametric surfaces, A_1 and A_2, corresponds to the geometry sketched in Figure 1 and is expressed as:

$$F_{1-2} = \frac{1}{\pi \int_t \int_s |\vec{n}_1| \ \ d_s d_t} \int_\theta \int_\phi \int_t \int_s \mathbf{f} \ d_s d_t d_\phi d_\theta (1)$$

where:

$$\mathbf{f}\,(s,t,\phi,\theta) = \frac{(\vec{n}_1.\vec{e}_{12})\ (\vec{n}_2.\vec{e}_{21})}{L^4}$$

$$\mathrm{L} = |\vec{e}_{12}| = |\vec{e}_{21}|.$$



t,s	=	parametric variables that describe A_1.
θ, ϕ	=	parametric variables that describe A_2.
$\vec{n}_1$	=	normal vector to A_1 in a point P_1.
$\vec{n}_2$	=	normal vector to A_2 in a point P_2.
$\vec{e}_{12}$	=	visibility vector from P_1 to P_2.
$\vec{e}_{21}$	=	visibility vector from P_2 to P_1.
α_1	=	angle between $\vec{n}_1$ and $\vec{e}_{12}$.
α_2	=	angle between $\vec{n}_2$ and $\vec{e}_{21}$.
L	=	distance between P_1 and P_2.

Figure 1: Geometry of the form factors.

In order to calculate the numerical value of F_{1-2}, the latter expression is evaluated using the numerical integration method known as Gaussian quadrature [Carnahan et al.69] , [Press et al.88]. This method consists, basically, of solving the integrals by changing them into summations. Considering the surfaces are divided into patches, the summation corresponding to the form factor between two patches **a** and **b** is given by:

$$F_{a-b} = \frac{1}{A_a} \sum_{i=1}^{p} \sum_{j=1}^{p} \sum_{k=1}^{p} \sum_{r=1}^{p} w_i w_j w_k w_r \; \mathbf{f}' \qquad (2)$$

where:

A_a	=	area of the patch **a**.
$\mathbf{f}'\ (s_i, t_j, \phi_k, \theta_r)$	=	$\frac{(\vec{n}_a.\vec{e}_{ab})\ (\vec{n}_b.\vec{e}_{ba})}{\lvert\vec{e}_{ab}\rvert^4}$
w_i, w_j, w_k, w_r	=	weight coefficients.
p	=	nr. of Gaussian points.

The function $\mathbf{f}'$ is evaluated applying the vectors $\vec{n}$ and $\vec{e}$ at sample points distributed over the patches. The coordinates of these sample points are given through the parametric equations that describe the surfaces.

In addition, the occlusion testing performed by PDM includes curvature tests which consider the vector characteristics of the calculation of the form factors. These tests influence directly the accuracy of the results and allow a reduction of the number of operations executed during the calculation of the form factors.

THE COOPERATIVE PROCESSES SOFTWARE

The CPS package is part of the *FERMILAB Advanced Computer Program* research and was primarily built for usage in *High Energy Physics*. It provides a variety of tools for the parallel application design, programming, debugging and execution. It is easy to run a CPS application on any local network of *UNIX* based workstations because of its emphasis on portability. Building and debugging a CPS application is also a quite simple task even in a heterogeneous network environment since it is possible to use the original host tools such as compilers, linkers, debuggers.

For application programming the CPS package provides a library of primitive routines which implements mechanisms such as synchronization, remote procedure call, and message transfer. To execute a parallel application the user needs to run the CPS *Job Manager*. This program starts the application processes on remote machines and then periodically checks their status.

There are three main steps for CPS application design. Initially, the application is decomposed into a set of cooperating tasks. Each task is functionally different from the others. The next step is the task programming. For each task the programmer must define a different source code, using the CPS library to add an interface to allow cooperation among tasks. Finally the programmer/user must create a *Job Description File*. This file describes the application execution. It contains information such as which processes are assigned to different processors and how many instances of each task (processes) should be created.

DESIGN AND IMPLEMENTATION OF THE MPC STRATEGY

Application Description

The application used in this paper to describe the MPC strategy consists of the calculation of the form factors regarding the environment presented in Figure 2 (see color plates). This environment represents a test model which is composed of a sphere interacting, through the light interreflections, with the six faces of a surrounding cube. The form factors are calculated using the PDM and the calculations are performed on a network composed of workstations SUN Sparc 2. During the execution of the calculations the network was not shared by others users.

The most important design step of a CPS parallel application is the partition of the sequential algorithm in a parallel solution composed by different tasks. The application used in this paper was divided into three tasks:

- *Server* task: executed by a set of N

replicas of the same program. Each of these replicas should be allocated to a different processor of the network. The *server* processes are dedicated to the execution of two services implemented through a remote procedure call mechanism: calculation of form factors face-face (**cff**) and calculation of form factors sphere-face (**csf**).

- *User* task: executed by one process that initializes the variables and loads the parameters files. The *user* process controls the execution of the services provided by the *server* processes through remote calls using the face(s) identification(s) as parameter(s).

- *Join* task: executed by one process that composes the form factors matrix using the results of the calculations performed by the *server* processes. Afterwords this matrix is used on the calculation of the patches radiosities following a full-matrix approach of the radiosity method.

The processing times regarding the sequential execution of the services are shown in Table 1. The services **cff** are the most critical, since they demand a higher processing time, especially for an environment divided into a very large number of patches.

Table 1: The sequential execution of the PDM.

Patches	Processing Time		
	csf	cff	Total
86	19s	42s	1m 3s
280	2m 33s	9m 48s	12m 23s
992	20m 33s	155m 57s	176m 35s

m=minutes ; s=seconds

The Services Granularity

The services **csf** and **cff** perform the calculations between pairs of patches that belong to different surfaces. The number of patch-patch form factors calculated per service invocation express the granularity of the service. It is essential to choose a granularity that fits better the application and the hardware characteristics in order to improve the application performance.

A coarse-grain service performs a large number of patch-patch form factors calculation per service invocation, exchanging relatively litle information between the involved processes. A fine-grain service performs few patch-patch form factor calculations per invocation but a larger number of invocations are necessary to perform the same job. Therefore, choosing an adequate service granularity implies the definition of a trade-off between the amount of messages exchanged between the *user* process and the *server* process and the amount of calculation performed at each service invocation.

There are several possible granularity levels for the services. However, taking into account that the environment is composed of surfaces which are divided into patches, we have selected three major levels:

- The coarse level: the services perform the calculations regarding the form factors between all pairs of patches which belong to two different surfaces.

- The medium level: the services perform the calculations regarding the form factors between a patch of a surface and all the patches of another surface.

- The fine level: the services perform the calculations regarding the form factor between a pair of patches.

The finest granularity level has become a prohibitive choice due to the high costs of interprocess communication in our loosely-coupled workstations network environment. Table 2 shows that for this level there is a large increase in the number of service invocations compared with the coarse-grain and the

medium-grain service implementations of the algorithm. However, the average overhead per service invocation remains constant regardless the granularity used. Therefore choosing the finest granularity level would lead our application to spend most of its processing time doing service invocation rather than calculating form factors.

Table 2: Granularities comparison (992 Patches).

Service Granularity	Number of Invocations	
	csf	cff
coarse	6	15
medium	768	2160
fine	110592	311040

Beyond this problem of interprocess communication overhead cost we must deal with the server process subutilization. This problem degrades the performance of coarser granularity services. It occurs when the total number of service requests is not multiple of the number of available processors on the network. In this case some processors will not be active all the time and this implies a loss in performance. Figure 3 illustrates the processes subutilization regarding 6 **csf** calculations and compares the 3-processors to the 4-processors case. An important issue to point out is that the processors subutilization is proportional to the service time and inversely proportional to the application time. This means that a coarser granularity implies longer service time and consequently higher subutilization rates. On the other hand, longer application time attenuates the subutilization problem.

The effects of the process subutilization problem can be observed in Figure 4. This figure shows the speed-up regarding the execution of our **csf** coarse-grain service implementation considering the environment divided in 992 patches. It is easily detected that the use of additional server processes does not necessarily improve the final speed-up (see the low speed-up improvement from the 3-processors to the 4,5-processors case).

The choice of the most suitable granularity is a difficult decision especially because minimizing interprocess communication overhead and subutilization rates at the same time are conflicting goals.

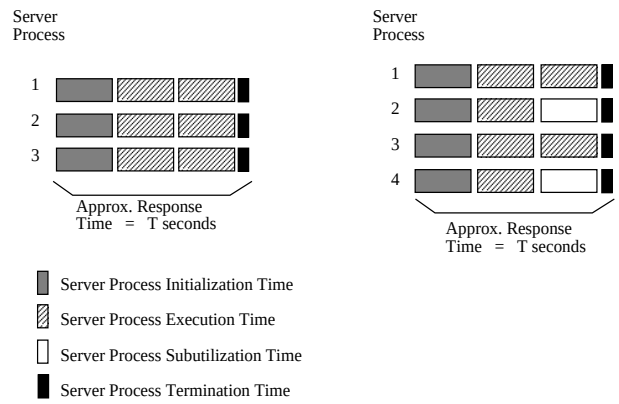


Figure 3: Processes subutilization.

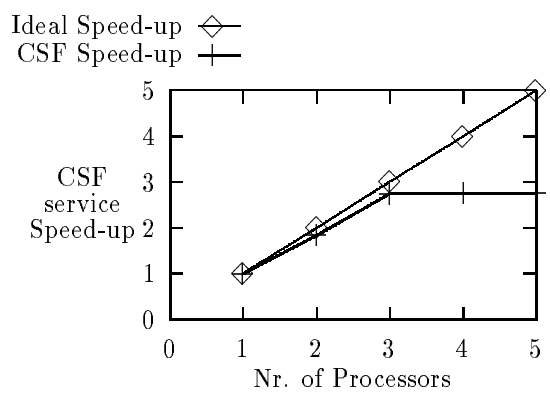


Figure 4: CSF Coarse-grain Speed-up.

The Hybrid Granularity Approach

One of our design decisions was that our algorithm would start with the execution of the **csf** service. Because of that, when there are no **csf** service requests to be carried out, the idle servers start immediately working servicing **cff** requests. This represents a partial execution overlap between **csf** and **cff** calculation that eliminates subutilization for the **csf** service. This interleaved execution can also introduce higher speed-up factors for the overall

application time than the ones achieved individually by the **csf** and the **cff** services.

However, for the **cff** service, which is the last one, no execution overlap is possible, thus subutilization becomes a critical problem. This can be observed in Figure 5 which correspond to an environment subdivision in 992 patches. This figure shows the comparison between coarse-grain and medium-grain speed-up factors for **cff**.

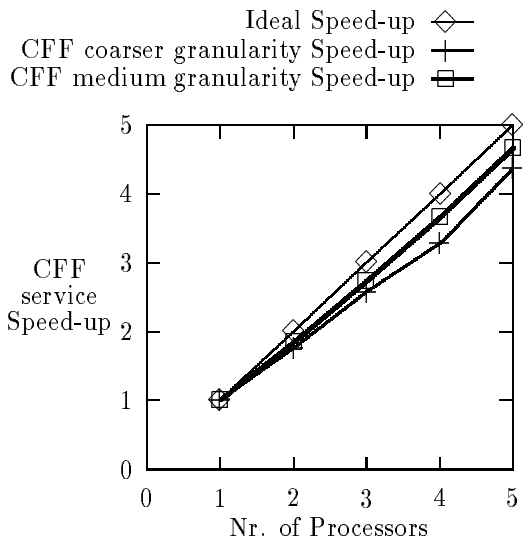


Figure 5: CFF service speed-up.

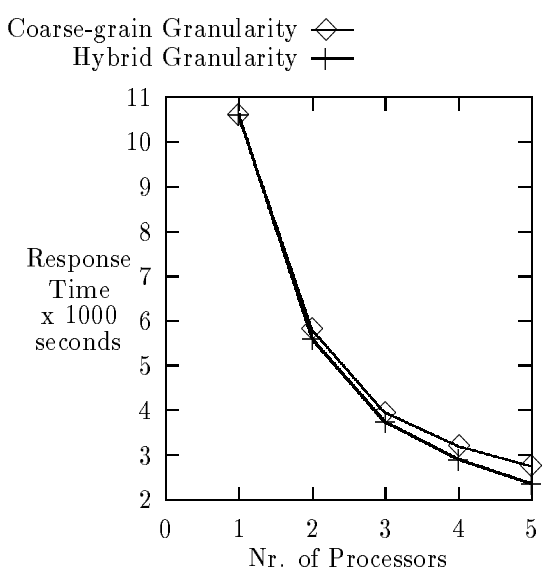


Figure 6: Application final response time.

A comparison between the final response time regarding the use of the coarse level of granularity for both services and the final response time regarding our hybrid approach is presented in Figure 6. The values presented in this figure are also related to 992 patches. This comparison makes clear the performance gain that can be achieved combining a coarse level for the **csf** service and a medium granularity level for the **cff** service.

RESULTS

The speed-up gain of the parallel over the sequential execution is sketched in Figure 7, taking into account the three subdivisions of the environment and using the hybrid approach described in the previous section. Through the analysis of this speed-up graph, we noticed that the application of the MPC strategy results in best performances for environments divided into a large number of patches.

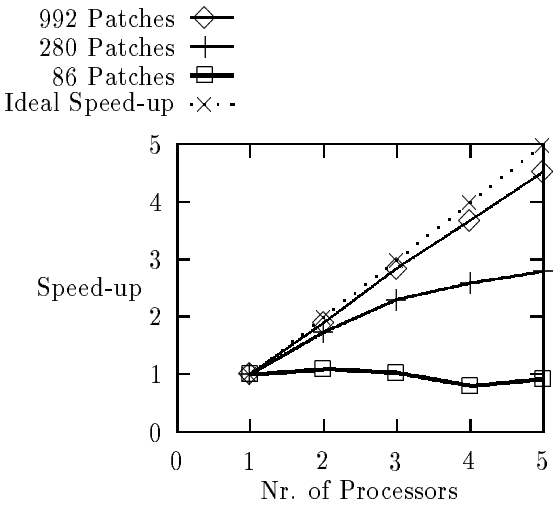


Figure 7: Speed-up x Number of Processors.

In fact, considering the subdivision of the environment into a small number of patches (for instance 86) the sequential algorithm performed better than the parallel version for the 4,5-processors cases. This is a consequence of a series of incurred overhead inherent to a parallel application such as remote process

initialization, interprocess communication, synchronization, general application management and remote process termination. For the 86 patches case these overhead costs override the achieved speed-up.

The overhead costs remain the same and the processing time increases when the environment is divided into a finer grid of patches. Therefore the overhead costs percentage in the total processing time becomes smaller as the number of patches increases. As a matter of fact, for the 992 patches per surface case we have almost reached the ideal speed-up limit. The processing times regarding the parallel execution of PDM, considering the use of five processors, are shown in Table 3. The comparison between these values and those presented in Table 1 emphasizes the processing time savings that can be achieved through the application of the MPC strategy.

Table 3: The parallel execution of the PDM.

Patches	Processing Time	Speed-up
86	1 m 9 s	0.92
280	4 m 27 s	2.78
992	39 m 4 s	4.52

m=minutes ; s=seconds

CONCLUSION

The obtained results demonstrate the speed-up gain of the parallel over the sequential execution of the PDM algorithm. However, as already mentioned, the use of additional server processes does not necessarily improves the performance. This happens due to two major problems: server process subutilization and interprocess communication costs. Both problems may be attenuated through the correct granularity selection for the services.

Future efforts will include the improvement of the PDM in order to perform the form factor calculation between vertices and patches. This procedure will improve the radiosities accuracy since it will allow their calculation directly in the vertices. Besides the calculation of form factors between vertices and patches will require the calculation of only one area integral instead of two. This simplification will reduce the processing time. In addition we will test another tool for parallel application design, HeNCE (Heterogeneous Network Computing Environment) [13], in order to compare its performance and design facilities against CPS ones.

ACKNOWLEDGMENTS

We would like to thank the research staff of LAFEX (Brazil) and Fermi National Accelerator Laboratory (FermiLab/USA) for allowing us the use of CPS. We are also grateful to Mr. Richard Cole (BBC/London) for the review of the draft version of this paper, to Mr. Rui Bastos (INESC/Portugal) for his comments and suggestions, to Prof. Anatolio Laschuk and Prof. Phillipe Navaux for their support and to our colleagues from CPGCC-UFRGS for the fruitful discussions.

REFERENCES

Baranoski,G.V.G. 1992 "The Parametric Differential Method: An Alternative to the Calculation of Form Factors", In: Proceedings of EUROGRAPHICS Conference and Exhibition, 7-11 Sept., Cambridge, UK, *Computer Graphics Forum*,11(3). pp. 193-204.

Baum,D.R., Rushmeier,H.E., Winget,J.M. 1989 "Improving Radiosity Solutions through the Use of Analytically Determined Form Factors", *Computer Graphics*, 23(3), pp. 325-334.

Beguelin,A., Dongarra,J.J., Geist,G.A., Manchek,R., Sunderam,V.S. 1991 *Graphical Development Tools for Network-based Concurrent Supercomputing*, In: Proceedings of Supercomputing 91, Albuquerque, USA, pp. 435-44.

Carnahan,B., Luther,H.A., Wilker,J.O. 1969 *Applied Numerical Methods,* John Willey.

Chalmers,A.G., Paddon,D.J. 1991 "Parallel Processing of Progressive Refinement Radiosity Methods", In: Proceedings of the Second EUROGRAPHICS Workshop on Rendering, 13-15 May, Barcelona, Spain.

Drucker,S.M., Schroder,P. 1992 "Fast Radiosity Using a Data Parallel Architecture", In: Proceedings of the Third EUROGRAPHICS Workshop on Rendering, Bristol, UK, pp. 247-58.

Feda,M., Purgathofer,W. 1991 "Progressive Refinement Radiosity on a Transputer Network", In: Proceedings of the Second EUROGRAPHICS Workshop on Rendering, 13-15 May, Barcelona.

Fermi National Accelerator Laboratory. 1989 "ACP Cooperative Processes Software - User's Manual", Universities Research Association.

Nash,T., Areti,H., Atac,R., Biel,J., Cook,A., Deppe,J., Edel,M., Fischler,M., Gaines,I, Hance,R., Husby,D., Isely,M., Miranda,E., Pham,T., Zmuda,T., Eitchen,E., Hockney,G., Mackenzie,P., Thacker,H.B., Toussaint,D. 1988 " High Performance Parallel Computers for Science: New Developments at the Fermilab Advanced Computer Program", In: Proceedings of the Workshop on Computational Atomic and Nuclear Physics at One Gigaflop, 14-16 April, Oak Ridge, USA.

Press,W.H., Flannery,B.P., Teukolsky,S.A., Vetterling,W.T. 1988 *Numerical Recipes in C The Art of Scientific Computing*, Cambridge University Press.

Wallace,J.R., Elmquist,K.A., Haines,E.A. 1989 "A Ray Tracing Algorithm for Progressive Radiosity", *Computer Graphics*, 23(3) pp. 315-325.