INTERACTIVE CONFIGURATION OF RESTRICTED SPACES USING VIRTUAL REALITY AND CONSTRAINT PROGRAMMING TECHNIQUES

Marouene Kefi[1,2], Paul Richard[1] and Vincent Barichard[2]
[1] *Laboratoire d'Ingénierie des Systèmes Automatisés (LISA), Université d'Angers, 62 Avenue ND du Lac, Angers, France*
[2] *Laboratoire D'Etudes et de Recherche en Informatique d'Angers (LERIA)*
Université d'Angers, 2 Bd Lavoisier, Angers, France

Keywords: Virtual environment, 3D interaction, Constraint programming, Multi-modal feedback, 3D-object layout.

Abstract: In this paper, we describe innovative approaches for the design of intelligent virtual environments (VE) for interactive problem solving. Thus, we propose to extend VEs to support constraint-based interaction through the use of Constraint Programming (CP) techniques. The aim of this paper is to argue for the need of CP integration in VEs and its high relevance in the 3D-objects layout problem. The user manipulation will be translated as incoming queries of the intelligent module (solver) which will generate a solution compatible with the design requirements. Thus, the task of the solver is to satisfy the constraints specified by the system in response to user interaction. In order to exhibit a high degree of visual richness and realism, the use of human-scale multi-modal 3D interaction techniques and tools is proposed.

1 INTRODUCTION

VEs are popular terms that refer to a variety of multi-sensorial computer-generated experiences. VEs technology is now recognized as a powerful design tool in industrial sectors such as manufacturing, process engineering, construction, and aerospace industries (Zorriassatine et al., 2003). However, in many cases, VEs are being used as pure visualization tools for assessing the final design (Drieux et al., 2005). One of the potential contexts in which VEs can be employed is decision making where complex tasks involving multiple criteria to be satisfied are to be achieved. For example, 3D-objects layout can be a time-consuming and tedious task. In most current systems, the user must position each object by hand using a computer mouse. In more advanced configurations, human-scale 3D interaction techniques and multi-modal feedback (visual, auditory, kinesthetic and tactile) may facilitate the placement of objects in the environment. However, the user has no indication concerning the best placement of objects that ensures satisfaction of constraints. Thus, VEs must be extended with advanced processing modules (such as constraint solvers, etc.) in order to assist the user in decision making.

The concept of constraint is naturally present in our everyday life. A constraint is defined as a logical relation between various unknown factors, called variables, each one taking its values in domain. Thus, a constraint restricts domains by removing values which can't be affected to the corresponding variables. The satisfaction of constraints requires a formalism which offers a structuring framework making it possible to model the problems expressed in terms of constraints. This formalism is called CSP (Constraint Satisfaction Problem). A CSP can be seen like a problem modeled in the form of a set of constraints on variables. The resolution of a CSP consists in assigning values to the variables, so that all the constraints are satisfied (Solnon, 2003).

Algorithms making it possible to solve a CSP are called constraints solvers. Some solvers have been integrated in programming languages, thus defining a new paradigm of programming called Constraint Programming (CP). To solve a CSP with a programming language by constraints, it is sufficient to specify the constraints, their resolution being automatically processed by the CP-based solver itself integrated into the programming language.
In this paper, we present a human-scale haptic VR platform that allow the user to position virtual ob-

Kefi M., Richard P. and Barichard V. (2010).
INTERACTIVE CONFIGURATION OF RESTRICTED SPACES USING VIRTUAL REALITY AND CONSTRAINT PROGRAMMING TECHNIQUES.
In *Proceedings of the International Conference on Computer Graphics Theory and Applications*, pages 384-389
DOI: 10.5220/0002893903840389
Copyright © SciTePress

jects in a 3D environment. Three approaches will be proposed to assist the user in decision making processes in the context of 3D-objects layout problems. These approaches make use of Constraint Programming (CP) techniques. The CP-based solver proposes solutions that satisfy all the constraints specified (offline or interactively) by the user.

In the next section, we present the previous work. Section 3 describes our human-scaled haptic VR platform. In section 4, we present our work concerning 3D interactive configuration of restricted spaces using our VR platform. In this case, the user is able to manually relocate the virtual objects within a VE. In section 5, we present more advanced approaches based on the use of CP techniques. The paper ends by a conclusion and provides some tracks for future work.

2 RELATED WORK

Preliminary works using CP or Constraint Logic Programming (CLP) in 3D environments has essentially been dedicated to the behavior of individual objects or autonomous agents within the environment. For instance, Codognet has included a concurrent constraint programming system into VRML to specify the behavior of artificial actors in dynamic environments (Codognet, 1999). Axling et al., (Axling et al., 1996) have incorporated "OZ" (Smolka et al., 1993), a high-level programming language supporting constraints into the DIVE (Andersson et al., 1993) distributed VR environment. Both Axling and Codognet have put emphasis on the behavior of individual objects in the virtual world and did not address user interaction or interactive problem solving. As demonstrated by Honda and Mizoguchi (Honda and Mizoguchi, 1995) and Pfefferkorn (Pfefferkorn, 1975), CP techniques are particularly appropriate for the resolution of configuration problems. Both have demonstrated the suitability of CP as an approach based on the declarative nature of the formalism which facilitates the description of the problem and its efficiency for avoiding combinatorial explosion.

More recent approaches for visualizing the execution of constraint programs in VEs have been developed. For example, Fages et al., have developed a generic graphical user interface (CLPGUI) for visualizing and controlling logic programs (Fages et al., 2004). The proposed architecture involves a CLP process and a Graphical User Interface (GUI) which communicate by sockets. Fernando et al., have exposed the design and implementation details of a constraint-based VE (Fernando et al., 1999). Xu et al., have treated the combination of physics, semantics, and placement constraints and how it permits to quickly and easily layout a scene (Xu et al., 2002). The layout task can be substantially accelerated with a simple pseudo-physics engine and a small amount of semantic information. Xu generalized distributions and a richer set of semantic information leading to a new modeling technique where users can create scenes by specifying the number and distribution of each class of object to be included in the scene. Calderon et al., have presented a novel framework for the use of VEs in interactive problem solving (Calderon et al., 2003). This framework extends visualization to serve as a natural interface for the exploration of configuration space and enables the implementation of reactive VEs. This implementation is based on a fully interactive solution where both visualization and the generation of a new solution are under the control of user. Sanchez et al. have presented a general-purposed constraint-based system for non-isothetic 3D-object layout built on a genetic algorithm (Sanchez et al., 2002). This system is able to process a complex set of constraints, including geometric and pseudo-physics ones. To get an easy-to-use object-layout software, they have described the 3D-scene by using semantic and functional features associated with the objects.
More recently, Jacquenot has developed a generic method to solve multi-objective placement problems for free-form components (Jacquenot, 2009). The proposed method is a relaxed placement technique combined with a hybrid algorithm based on both an evolutionary algorithm and a separation algorithm. Moreover, different elements for solving 3D regular and complex geometry problems have been also proposed.

It must be noted that these previous works are based on CLP and Prolog (Diaz and Codognet, 2001) or genetic algorithms. However, in the last few years, powerful CP-based solvers such as Gecode (Schulte, 1997), CHOCO (Jussien et al., 2009), ILOG CP (IBM,) have been developed. Unlike CLP and Prolog, these CP-based solvers are designed as library. In addition, they provide an API to developers to embed constraint programming in another program written in an object language (C++ or Java). Moreover, they are not dependent on logic programming and make it easy to use constraints in an independent and efficient solving engine.

The development of CHOCO started in 1999 within the OCRE project, a French national initiative for an open constraint solver for both teaching and research applications. CHOCO is an open source java library for CSP and CP. It is built on an event-based propagation mechanism with backtrackable structures. Gecode is also an open source, free, portable, accessible, and efficient environment for developing constraint-based systems and applications. Gecode can be easily interfaced to other systems.It supports the programming of new propagators (as implementation of constraints), branching strategies, and search engines. New variables can be programmed at the same level of efficiency as integer and set variables that ship. Another very powerful CP-based solver is IBM ILOG CP Optimizer (commercial tool). It uses CP to solve detailed scheduling problems and combinatorial problems not easily solved using mathematical programming methods.

Our approach, based on Gecode extends the work of Calderon et al. in different directions. For instance, in terms of mechanisms of user interaction, we envisage to offer more interactivity to the user for more efficient object manipulation (Bukowski and Squin, 1995; Kallman and Thalman, 1999; Stuerzlinger and Smith, 2000). As well, taking advantage of the incremental capabilities of the CP-based solver, we will give the user the possibility of adding objects and to select the constraints for those objects from a set of predefined constraints and to provides more feedback from the configuration. An explanatory module that would provide the user for justifications for the proposed solutions will be envisaged. Such a module is required to explain why there is no acceptable solution for some object positions proposed by the user.

3 HUMAN-SCALE HAPTIC VE

This section presents the human-scale VE called VIREPSE that provides force feedback using the SPIDAR system (Space Interface Device for Artificial Reality) (Richard et al., 2006). Stereoscopic images are displayed on a rear-projected large screen (2m x 2.5m) and are viewed using polarized glasses. In order to provide force feedback to the user's hands, four motors are placed on the corners of a cubic frame surrounding the user. By controlling the tension of each string, the system generates appropriate forces.

3.1 System Workspace

VIREPSE workspace could be divided into two spaces: (1) reachable space that gathers every point users can reach with hands, and (2) haptic space that gathers every point where the system can produce a force in any direction (Fig. 1).

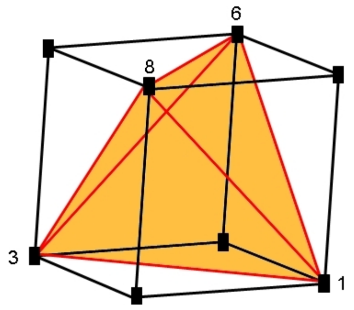

Figure 1: Illustration of the system workspace.

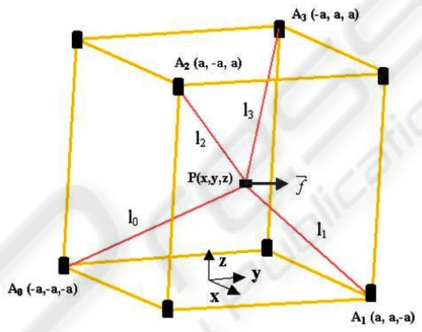

Figure 2: Framework for the application of contact forces on the user hand.

3.2 Position Measurement

Let the coordinates of the user's hand position be $P(x,y,z)$, which represent both the hand position, and, l_i, the length of the i^{th} string $(i = 0,1,2,3)$. Let the four motors be on four non-adjacent vertices of the SPIDAR cubic frame. Then, P(x,y,z) must satisfy the equations (1) as illustrated in Fig. 2.

$$\begin{cases} {l_0}^2 = (x+a)^2 + (y+a)^2 + (z+a)^2 \\ {l_1}^2 = (x-a)^2 + (y-a)^2 + (z+a)^2 \\ {l_2}^2 = (x-a)^2 + (y+a)^2 + (z-a)^2 \\ {l_3}^2 = (x+a)^2 + (y-a)^2 + (z-a)^2 \end{cases} \tag{1}$$

Let the length of the SPIDAR cubic frame be 2a (Fig. 2). After some mathematical manipulations, we can obtain the position of the user's hand in function of the lengths l_i (equations 2) :

$$\begin{cases} x = \frac{{l_0}^2 - {l_1}^2 - {l_2}^2 + {l_3}^2}{8a} \\ \\ y = \frac{{l_0}^2 - {l_1}^2 + {l_2}^2 - {l_3}^2)}{8a} \\ \\ z = \frac{{l_0}^2 + {l_1}^2 - {l_2}^2 - {l_3}^2)}{8a} \end{cases} \tag{2}$$

Let the resultant force felt by the operators be f and the unit vector of the tension be $\overrightarrow{u_i}$, $(i = 0,1,2,3)$. f is given by equation 3.

$$\overrightarrow{f} = \sum_{i=0}^{3} k_i \overrightarrow{u_i}, k_i > 0 \qquad (3)$$

k_i represents the tension value of each string.

4 3D-OBJECT LAYOUT WITHOUT CP

In this section we describe how 3D-object layout is currently achieved using our human-scale VE (without CP). Although this approach may be viable for 3D-object layout in simple environments with very limited and non-demanding constraints, it cannot be applied in more complex situations involving many objects and constraints. As illustrated in Figure 3, the user selects and moves 3D-objects in space using a direct manipulation technique. To select a given object, the user has to place a 3D cursor (small cube) inside the volume of this object and close his/her hand. A dataglove is used to detect user's hand configuration (open or closed). Then, the user's hand movement is mapped to the movement of the selected object. In order to increase the system accuracy and to allow the user to rotate the 3D-objects is space, a motion capture system is used. Thus, the dataglove was equipped with a set of reflective markers (grey circles in Fig. 3). In order to improve depth perception through real-time head tracking (motion parallax), a reflective marker was placed on a cap worn by the user. Multi-modal feedback is used to prevent the user to move beyond the limits of the layout environment and to help him to correctly layout the objects.

5 INTERACTIVE APPROACHES USING CP

Our system will aim to be real-time 3D environment based on Constraint Programming (CP) that support interactive problem solving and produce solutions within a time frame matching that of user interaction. Through communication with threads, user interaction and user constraints choice will be converted in real-time into appropriate CP-solver queries which will be translated back into automatic reconfigurations of objects in the VE. In others words, interacting with objects in the VE and selecting the constraints for that object will serve as inputs to the CP-based solver which will output new solutions for object configurations.

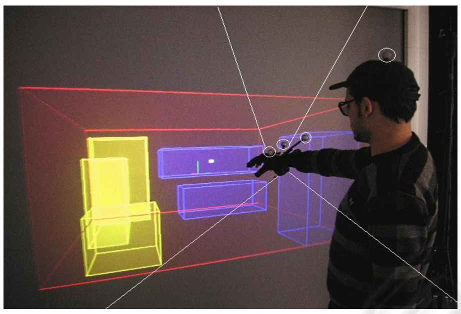

Figure 3: Interactive 3D-object layout using the human-scale stringed-based haptic virtual environment(without CP).

In addition, this system will allow to present to the immersed user several solutions (feasible configurations) that he/she will be able to explore.
The solving mechanisms of many constraint systems will be triggered by any modification of variable values or/and constraints. In our case, user interaction with the virtual objects will be translated into input to the CP-based solver by selecting the variables whose values have been altered by the interaction and adding the constraints whose have been chosen from the constraints menu. For instance, when visualizing a configuration of objects, the user can alter the position of certain objects which modify the constraints involving these objects. This triggers the CP-based solver on a new set of variables and constraints. The solver in turn outputs resulting solutions in the form of new object configurations within the 3D environment. The user will then be able to freely interact with objects. Solution space exploration will start by proposing a first solution (configuration of objects) and will allow the user to explore other possible configurations. Once the user has selected a configuration, he will be able to interact with it by displacing the constituent objects.

5.1 New Approaches

Once the 3D objects and the constraints have been selected by the user, the system will begin geometrical computing with the aim to verify if the selected objects can be placed within the environment. Then, the user will have three possibilities for interaction with the 3D-object configuration, illustrated in Fig. 6, Fig. 4 and Fig. 5, and described the following paragraphs.

First Approach. As a response to constraints and objects selection, a feasible configuration will be computed by CP-based solver and displayed within the VE. In this case, only solution visualization will be proposed. Thus, the user will not be able to interact with the VE.

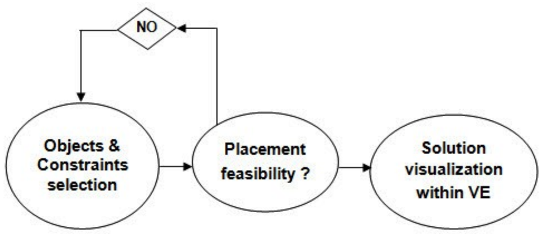
NO
Objects & Constraints selection
Placement feasibility ?
Solution visualization within VE

Figure 4: Illustration of the first approach.

Second Approach. It is similar to the first approach, but the user cans interact with the proposed solution by displacing its constituent objects. For more immersion and accurately, the system provides multi-modal feedback such as visual, auditory and kinesthetic. In this case, the problem-solving module will not be called to compute a new solution but to verify the new set of placement constraints corresponding to the user interaction. However, and taking into account the new allocation of displaced object (so new constraints),the system will react according to this last displacement as follows :

- If the CP-Solver can generate a new solution with the new set of constraints, the VE will update itself and proposes that solution.
- If there is no solution that response to the new user-displacement, the system will cancel the last displacement and gives justifications on its unfeasibility.

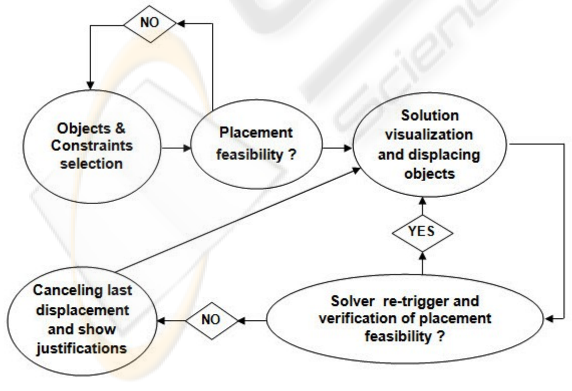
NO
Objects & Constraints selection
Placement feasibility ?
Solution visualization and displacing objects
YES
Canceling last displacement and show justifications
NO
Solver re-trigger and verification of placement feasibility ?

Figure 5: Illustration of the second approach.

Third Approach. The system enables the user to add objects one by one using a specific interaction device. This in turn will disrupt the imposed constraints and forces the system to propagate all the constraints then generate a new solution compatible with the design requirements. Thus, the system automatically reconfigures itself as a consequence of addition of a new object. Multi-modal feedback are also used.

After each addition, the user is still allowed to modify manually the objects layout as described in the two previous sections (first and second approaches). It is important to specify that user interaction will not be translated to a new 3D objects layout problem, but its only add new constraints to the set of constraints already defined. Thus, each addition of new object will add new constraints to the initial ones. In the future, a particular attention will be devoted to extend our system to support user removing constraints and /or objects. We will embed Gecode in our programming environment because it allows the implementation of many different types of constraints. This makes possible to represent *semantic constraints*, i.e. constraints involving object properties such as materials, friction coefficient, resistance to fire, etc. Moreover, Gecode is an open source tool which on our team have a solid experience. However, we will use both open source tools (Gecode and CHOCO) in our scientific approach.

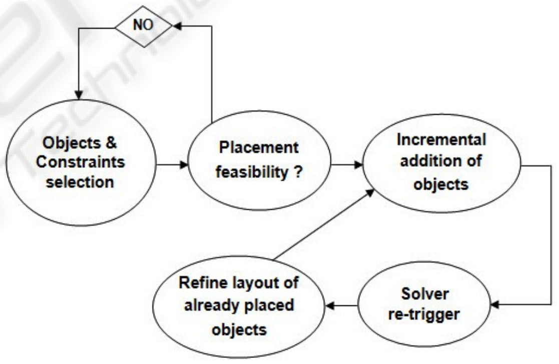
NO
Objects & Constraints selection
Placement feasibility ?
Incremental addition of objects
Refine layout of already placed objects
Solver re-trigger

Figure 6: Illustration of the third approach.

6 CONCLUSIONS

We have presented a human-scale multi-modal virtual environment (VE) that has been developed for basic computer-aided design purposes. The user selects and manipulates 3D objects in space. Haptic feedback is provided using a large-scale string-based device. In complex situations involving 3D-objects layout in large constrained environments, artificial intelligence techniques are required. We argue for the integration of Constraint Programming (CP) techniques based on recent and very efficient CP-based solvers taking into account discrete and continuous constraints simultaneously. Our proposition is based on the connection of constraints selection and user interaction which are taking place in the environment to the inputs/outputs of the solver. Thus, any user modification of objects position within the environment will update the constraints and trigger the solver that will in turn output resulting solutions in the form of new object configurations within the 3D environment, once their placement constraints have been set. In addition, we envisage to widen our research for providing the user with more explanation in order to justify why there exist no acceptable solutions for some object locations proposed by the user.

REFERENCES

Andersson, M., Carlsoon, C., Hagsand, O., and Stahl, O. (1993). Dive the distributed interactive virtual environment, tutorials and installation guide. In *Swedish Institute of Computer Science*.

Axling, T., Haridi, S., and Fahlen, L. (February 1996). Virtual reality programming in oz. In *Proceedings of the 3rd EUROGRAPHICS Workshop on Virtual Environments*.

Bukowski, W.-R. and Squin, H.-C. (1995). Object associations. In *ACM Symp. On Interactive 3D Graphics, Monterey, CA, USA*.

Calderon, C., Cavazza, M., and Diaz, D. (2003). A new approach to the interactive resolution of configuration problems in virtual environments. *Lecture notes in computer science*, 2733:112 – 122.

Codognet, P. (1999). Animating autonomous agents in shared virtual worlds. In *Proceedings DMS'99, IEEE International Conference on Distributed Multimedia Systems*. IEEE Press.

Diaz, D. and Codognet, P. (2001). Design and implementation of the gnu prolog system. *Journal of Functional and Logic Programming*, Vol. 2001, No 6.

Drieux, G., Guillaume, F., Leon, J., and Chevassus, N. (2005). Samira: A platform of virtual maintenance simulation with haptic feedback incorporating a model preparation process. *Proceedings of Virtula Concepts*.

Fages, F., Soliman, S., and Coolen, R. (2004). Clpgui: A generic graphical user interface for constraint logic programming. *Constraints*, 9:241 – 262.

Fernando, T., Murray, N., Tan, K., and Wimalaratne, P. (1999). Software architecture for a constraint-based virtual environment. *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 147 – 154.

Honda, K. and Mizoguchi, F. (1995). Constraint-based approach for automatic spatial layout planning. In *Conference on Artificial Intelligence for Applications*. IEEE Press.

IBM. *ILOG Products and solutions, http://ftp.ilog.fr/products/cp/*.

Jacquenot, G. (2009). Mthode gnrique pour loptimisation dagencement gomtrique et fonctionnel. *Thse de Doctorat, Ecole Centrale de Nantes*.

Jussien, N., Prudhomme, C., Cambazard, H., Rochart, G., and Laburthe, F. (2009). *choco: an Open Source Java Constraint Programming Library*.

Kallman, M. and Thalman, D. (1999). Direct 3d interaction with smart objects. In *ACM International Symposium on Virtual Reality Software and Technology, London. UK*.

Pfefferkorn, C. (1975). A heuristic problem solving design system for equipment or furniture layouts. *Communications of the ACM*, 18(5):286–297.

Richard, P., Chamaret, D., Inglese, F., Lucidarme, P., and Ferrier, J. (2006). Human-scale virtual environment for product design: Effect of sensory substitution. *The International Journal of Virtual Reality*.

Sanchez, S., Roux, O. L., Inglese, F., Luga, H., and Gaildart, V. (2002). Constraint-based 3d-object layout using a genetic algorithm.

Schulte, C. (1997). Oz explorer: A visual constraint programming tool. *Proceedings of the Fourteenth International Conference on Logic Programming*, pages 286–300.

Smolka, G., Henz, M., and Wurtz, J. (1993). Object-oriented concurrent constraint programming in oz. research report. In *Deutsches Forschungszentrum fur Kunstliche Intelligenz*.

Solnon, C. (2003). Programmation par contraintes. *http://www710.univ-lyon1.fr/ csolnon/Site-PPC/e-miage-ppc-som.htm*.

Stuerzlinger, W. and Smith, G. (2000). Efficient manipulation of object groups in virtual environments.

Xu, K., Stewart, J., and Fiume, E. (2002). Constraint-based automatic placement for scene composition. In *Graphics Interface Proceedings, University of Calgary*.

Zorriassatine, F., Wykses, C., Parkin, R., and Gindy, N. (2003). A survey of virtual prototyping techniques for mechnanical product development. *Journal of Engineering Manufacture*, 217-part B.