27

REAL-TIME COLLABORATIVE DESIGN SYSTEM FOR PRODUCT ASSEMBLY OVER THE INTERNET

Xiangxu Meng, Weiwei Liu, Yanning Xu
Dept. of Computer Science and Technology, Shandong University, Jinan, P.R. China
mxx@sdu.edu.cn; lww@mail.sdu.edu.cn; xyn@sdu.edu.cn

Product assembly design is a complex activity possible involving collaboration between different designers geographically dispersed. This paper puts forward some significant methodologies and technologies for distributed assembly and presents a collaboration architecture that manages to support working in both synchronous and asynchronous ways. Meanwhile, we adopt a three-level conflicts detection scheme to avoid conflicts effectively and a data streaming technology based on C/P (command/ parameter) to realize the real-time design. Based on the technologies mentioned above, a design system that supports real-time collaborative assembly is developed and we validate it by assembling a mechanical press across network collaboratively.

1. INTRODUCTION

Product assembly plays a critical role in getting products with high quality. However, as to many large and complex products such as mechanical press (Figure 1), designers generally need to assemble and debug all physical parts of products beforehand in the factory, then disassembling all parts for transportation, and at last assemble the parts again in the application place, which results in the great waste of both manpower and material resources. That is why we need a collaborative assembly design system for long-distance guidance and discussion to reduce the time of assembly. Meanwhile it is also propitious to make different domain designers participate in the assembly process planning.

Figure 1 – Mechanical press's pre-assembly in factory

Collaborative assembly design is an important application field of CSCW, however, the current CAD software mainly serves to planar

drawing or three-dimensional modeling, and adds strict restrictions to users that there must be only one person manipulating the object at the same time. Such typical software are: SolidWorks eDrawing，Hoops/Net, Actify SpinFire etc, and all of them can’t meet the collaborative assembly requirements.

There are many researches about collaborative design up to now: (Li, 2005; Gao, 2004) gave a general summary about the research and development status of collaborative computer-aided design. (Shyamsundar, 2001) presented a new geometric representation called AREP and a prototypical system (cPAD) (Shyamsundar, 2002) based on B/S architecture which, however, couldn’t support real-time collaborative work. (Rajarathinam, 2000) studied how to represent a geometric modeling in a multi-modal multi-sensory virtual environment supporting collaborative design, but didn’t talk about how to avoid conflicts in the collaborative condition.

To realize a collaborative assembly design in real time, there are still many technologies worth to studying such as: what kind of architectures should be adopted, “Thin server+ strong client” or “Strong server+ thin client”; how to control conflicts during the collaborative manipulations; how to transmit 3D streaming quickly and so on. We studied some key technologies about the real-time assembly on the premise that “if a product can be disassembled, it can be assembled too.” and a prototype system is developed supporting remote designers to evaluate assembly planning in real time over the network simultaneously.

2. DISTRIBUTED SYSTEM ARCHITECTURE

At present, there are mainly two categories of distributed system architectures: “Thin server+ strong client” and “Strong server+ thin client”. In the first architecture, clients are equipped with whole CAD functions and some communication facilitators. A server plays as an information exchanger to broadcast CAD files or commands generated by a client to other clients. Obviously, easy-deployed is the most straightforward character of this architecture through equipping standalone CAD systems with a communication facility; however, its conflicts are difficult to control without enough information. In the second architecture, the data structures in clients are light-weight and they primarily support visualization and manipulation functions. The main modeling activities are carried out in a common workspace in the server side, which is getting more popular since it brings a new kind of business model—application service provider (ASP). Nevertheless the network bandwidth is heavy-weight, making the delay of communication serious and the server apt to becoming a bottle-neck.

Considering the functional and performance requirement, we extend the functions of both server and clients based on the “Thin server+ strong client” architecture, and put forward a new infrastructure– composite framework (Figure 2). The usage scenario is as follows: when one client loads a product model (module 3) to a new session (module 1, 2), the system will transmit the model with its feasible disassembly sequences which are calculated beforehand (module 4) up to the server automatically. The other clients join the session (module 1, 2) and download specific geometric information (module 3) accordance with specific situations, say network status or different requirements. Then designers can start assembly planning

(module 6, 7) in the same virtual environment with various interactive manners (module 5) such as the mouse、keyboard、microphone、Flock of Birds、CyberGlove etc. Meanwhile the assembly route (module 8) reflecting the designer's interactive operations is recorded automatically. During the above process each client just only transmits those orders that have changed the assembly scene to the order queue (module 9) in the server. And the server takes out one order from the queue ordinally and judges if it will bring a conflict (module 10, 11). After making sure the order will not, the server transmits it forward to the other clients (module 12) where orders are parsed and executed in succession (module 6,7). Different clients can communicate with each other expediently through a chat channel.

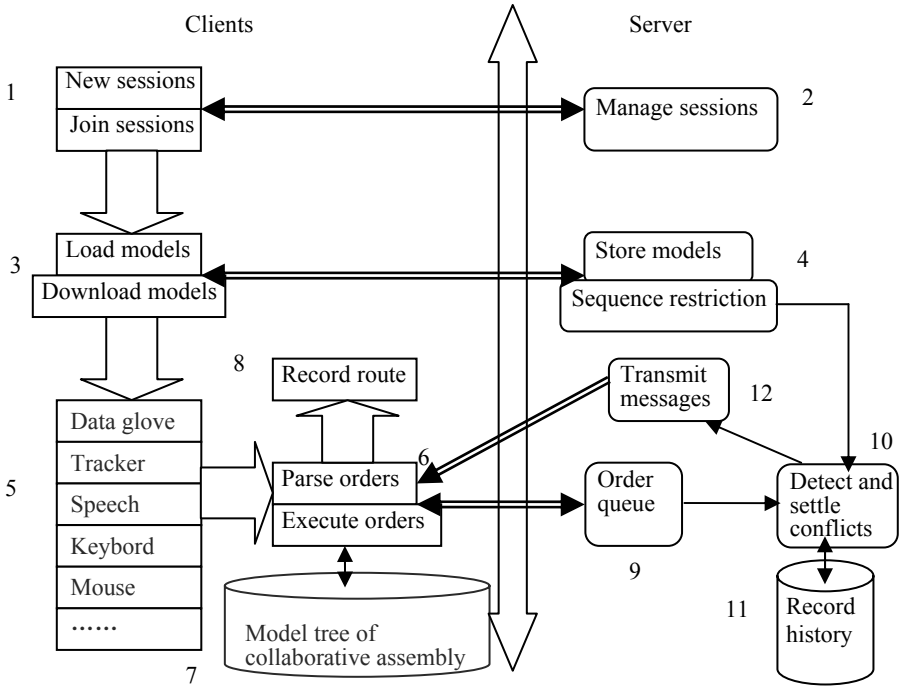

Figure 2 – Architecture of the composite framework

Compared to the typical architecture of "Thin server+ strong client", the composite framework adds several detecting modules to the server which succeeds in controlling conflicts in good time and enhancing the intercurrent ability; meanwhile, it lights the weight of clients by virtual of transferring some functions' execution to server, for instance models storage. In addition, adding the history record module in the server is propitious to working asynchronously or resuming from fault through executing orders from the records in succession. The system is developed from HOOPS 3D and HOOPS/NET, which is owned by the company Tech Soft and supplies many basic collaborative functions and manipulations as object selection, translation, collision detection, view point transform etc. All of those contribute to the implement of our further research and application.

3. CONFLICTS CONTROL MECHANISM

Team management and design coordination functions are the crucial components for establishing a well-organized team and avoiding conflicts on the whole. A working session mechanism is effective for the team management. Each session can be used to organize a collaborative task, and designers in the same session can share the design information dynamically. In a system, different design tasks can be carried out at the same time in different sessions.

Even so, the conflicts can't be avoidable completely because some design tasks must be accomplished by collaboration. There are two primary causes leading to conflicts in the system: incorrect local manipulations and sharing resources. To avoid those conflicts we must pay attention to two aspects: control-class and domain-class (Li, 2002), the former mainly deals with what mechanism is effective to avoid conflicts on the whole, while the latter mainly deals with how to manage the conflicts appeared already. Considering the complexity and diversity of conflicts, a three-level conflicts detection scheme is developed adopting different strategies accordance with different levels, which works effectively via filtrating conflicts level by level (Figure 3).

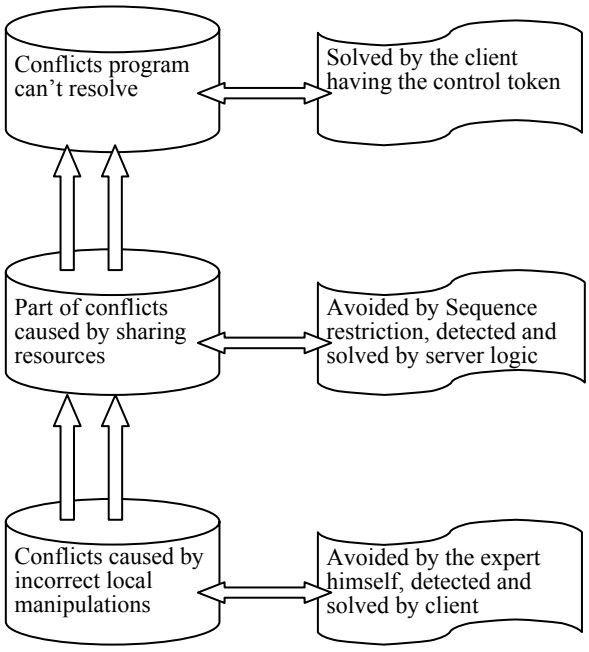

Figure 3 – Three-level conflicts detection scheme

(1) Conflicts caused by incorrect local manipulations. Each part has its own restrictions, for example disassembly direction and tool kits etc, all of which have been stored in the corresponding part restricts table locally. The disassembly actions of an expert are decomposed into many basic manipulations for instance selecting, holding, translating, revolving, releasing and so on. The system automatically

detects conflicts by virtue of the corresponding part restricts table while an expert is operating an object. If any conflict is found, the system will command the "conflicts settling module" to execute an "Undo" order with sending a warning message to the user.

(2) Part of conflicts caused by sharing resources. Due to the large quantity and complex relationships of parts, we can imagine how difficult it is to judge whether the current part can be disassembled or not, that will bring the process of cooperative manipulations into a disordered state and easily induce conflicts. To resolve this problem, We add one "sequence restriction module" to the server, which serves to calculate which parts are knock-down at the present time and change those parts' color into black as hints. So designers now can select and manipulate parts with the assistance of the sequence restriction orderly. In addition, associated with the current disassembly situation and history record, every order likely to cause conflicts must be compared with the conflict classes in the program. If any conflict takes place, program logical module responses to resolve it utilizing correlative domain knowledge and current data.

(3) Conflicts that program can't resolve. The system adopts control token mechanism: there are two classifications in users, one of them with control token and the others not; the client loading models has the control token as default which can be transferred by application. In the situation when there happened those conflicts even the program logic can't resolve, the system will clew users discussing with each other, and finally the client with control token has the most powerful word to resolve the conflicts.

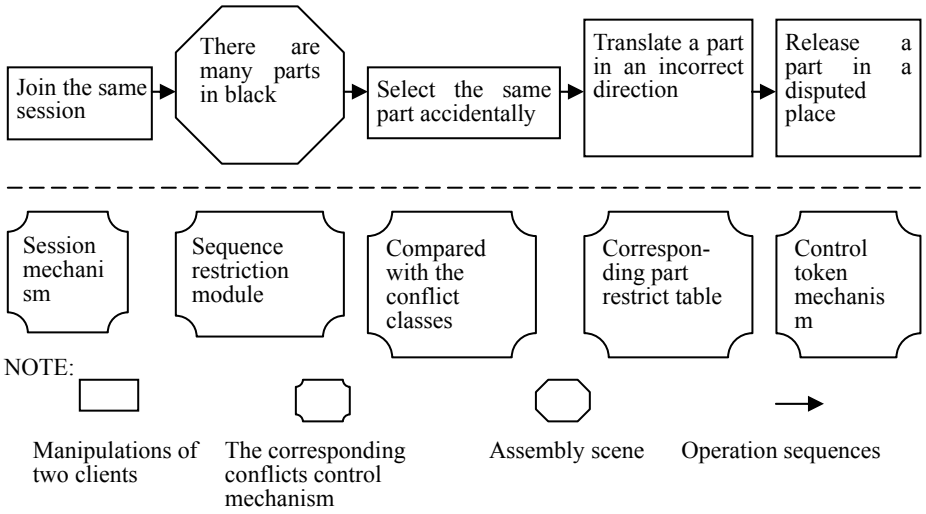

Figure 4 – An example of conflicts control process

Let's take an example to see how conflicts are controlled in the practical collaboration (Figure 4): Suppose that two clients A and B join the same session for carrying out a disassembly task, after joined in there are many parts in black to assist designers operating in the virtual scene. Client A selects one black-color part, written into the history record module immediately, and at one time client B also selects the same part accidentally, which causes a conflict can be easily found out by

checking the history record and comparing with the conflict classes in the program. Then program logical module suggests that client B select another part to avoid the conflict. Having selected two different parts, the clients can disassemble objects separately. At the time when one client is translating the part in an incorrect direction, a collision will take place sprung by searching the restricts table of the corresponding part which records the correct disassembly direction of each part, and the program executes an "Undo" order to resume the part's original position. At last, when a part is released in one place where other clients don't agree, the client with control token will make a conclusive decision according to the actual situation.

4. REAL-TIME SYNCHRONOUS COLLABORATION AND DATA TRANSMISSION

Multi-users' assembly platform calls urgently for real-time collaboration, but a significant problem for the above systems is that communication efficiencies are still quite far from satisfactory when large-size feature- and assembly-based models are designed collaboratively. Despite of many studies about effective 3D streaming transmission such as data compression, mesh simplification and incremental transmission, they can't satisfy the real-time requirements.

Representation of models should take into account of practical needs to decide which kind of representation is better. Notice that there are two primary points in our application: collaboration and assembly, deciding that the representation of models should not only care about the speed of network, but also should be convenient for assembly manipulations. In fact there are a mass of data in CAD model not needed in assembly, for example measurement data etc, so our models only keep down those information relevant with assembly, involving: many attributes information such as geometry, lamp-house, view point etc to show how to draw models; and some necessary restrictions information such as disassembly direction, disassembly tools etc to assist assembly simulations. Model tree structure is in favor of assembly manipulation. If someone manipulates a sub-assembly for instance translating or detecting collisions, it will be convenient to deal with by calculating all the sub-nodes of the sub-assembly, changing their correlative attributes, finding out which parts should be redrew, and sending these information forward to the hardware which redraws the scene accordingly.

Although we have predigested models, the amount of data is still large relative to the Internet with limited bandwidth capability. In order to reduce the load of network and increase the speed of transmission, the system adopts a new data streaming technology based on C/P (command/ parameter), i.e. the model data are transmitted only once after clients join a new session, and they will not be transmitted any more in the whole process. The system only transmits commands and parameters changing the disassembly scene to the other clients, and then the clients parse and execute those orders by themselves (Figure 5). Through using this technology we decrease the amount of transmission data and make real-time collaboration a reality.

After clients download model data to their own computers, they can start carrying out disassembly experiments by interaction. There are three kinds of interactive manipulations: the first kind is view point transform genus, for example

revolving view points or zooming the scene; the second kind is scene update genus, for example selecting, translating or revolving parts; the third kind is communication genus, for example sending chat messages to one another etc. Because every client manipulates different parts with the different corresponding view points, there is no need to send view point messages to the others; communication messages can be transmitted to each other directly; as to the scene update genus which changes the current disassembly state, it's necessary to send messages in C/P mode to tell other clients changing their own model trees. The corresponding relations are demonstrated in Figure 5.

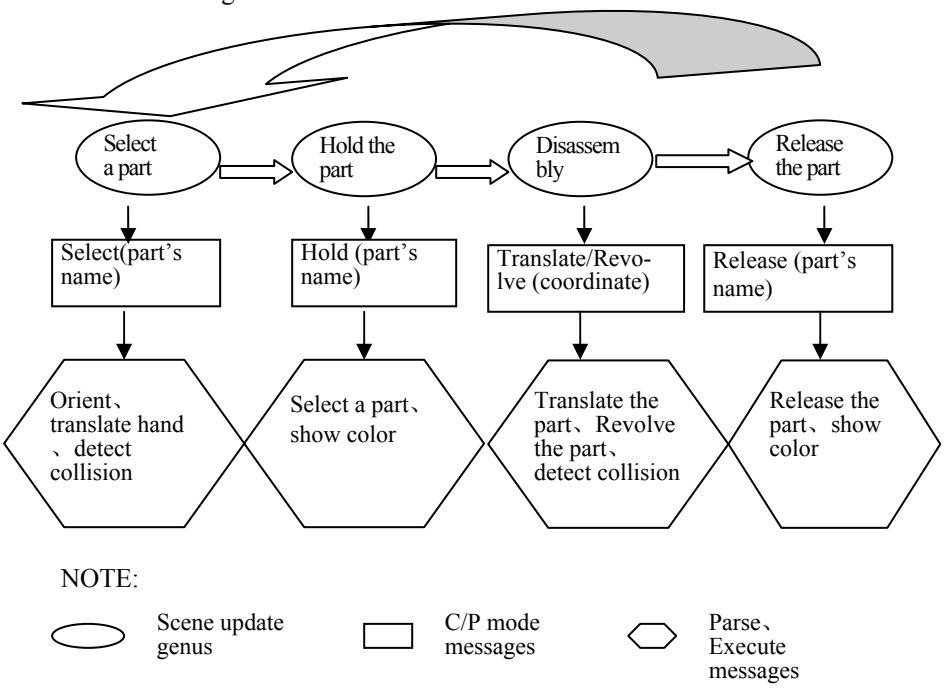

Figure 5 – Corresponding relations between clients' manipulations and transmitted messages

5. CONCLUSIONS

This paper studies some key technologies of collaborative assembly, involving framework of system, representation of models, transmission of data, avoidance of conflicts etc, and produces a real-time collaborative working platform proved availability by disassembling a mechanical press in the different places simultaneously (Figure6, 7). Further work would include how to translate different quantity of information to users according to their specific requirements and how to supply more convenient channels to communicate with each other effectively.

Figure 6 – Collaborative work by the designers from different domains with multimodal interaction(speech,data glove and tracker)

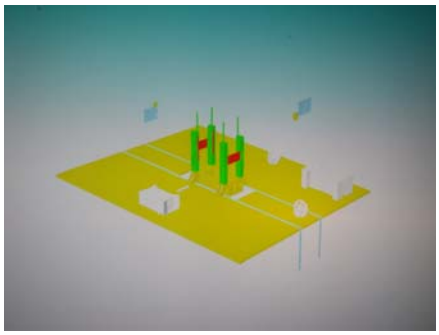

Figure 7 – Scene of collaborative disassembly (every hand stands for one client, the parts in black are knock-down for the moment.)

6. REFERENCES

1. Shuming Gao, Fazhi He. Survey of distributed and collaborative design. Journal of Computer-Aided Design & Computer Graphics, 2004; 12:149-157.1.
2. Yuemei Li, Shouqian Sun, Yunhe Pan. Multi-agent conflict resolution in cooperative design. Journal of Computer-Aided Design & Computer Graphics, 2002; 2:158-162.
3. W.D. Li, W.F. Lu etc. Collaborative computer-aided design – research and development status. Computer-Aided Design, 2005;37:931-940.
4. Rajarathinam Arangarasan, Rajit Gadh. Geometric modeling and collaborative design in a multi-modal multi-sensory virtual environment. Proceedings of DETC'00. ASME 2000 Design Engineering Technical Conference and Computers and Information in Engineering Conference, Baltimore, Maryland, September, 2000;10-13.
5. N.Shyamsundar, R.Gadh. Internet-based collaborative product design with assembly features and virtual design spaces. Computer-Aided Design, 2001; 33:637-651.
6. N.Shyamsundar, R.Gadh. Collaborative virtual prototyping of product assemblies over the Internet. Computer-Aided Design, 2002;34:755-768.