BETTY: Planning and Generating Animations for the Visualization of Movements and Spatial Relations

Andreas Butz, German Research Center for Artificial Intelligence Inc.,
D-66123 Saarbrücken, butz@dfki.uni-sb.de

Abstract

In this work the problem of synthesizing an animation is regarded as a planning problem, and the result is an animation planner that, starting from a visualization goal, plans a script for an animation including all low-level camera and object motions. The system takes all decisions about camera positions, zooms, moves and cuts considering the actual context as well as some fundamental filmmaking rules. The script is then realized by a given animation system which computes the single frames and does the playback.
BETTY is part of the multimodal user interface WIP, that investigates the plan-based automatic generation of multimodal operation instructions for technical devices.
Presently, BETTY is able to compute animations to demonstrate movements, to localize parts of a device and to explode assemblies.

Animation in Multimodal Documents

The project WIP (Knowledge-based Presentation of Information [Wa92, Wa93]), in which this work was done, investigates the coordinated presentation of information in several presentation modes at a time under given restrictions and circumstances (presentation medium, user model, restricted resources). The WIP system generates multimodal operation instructions for technical devices from geometric models and symbolic representations of these devices and the operations that can be performed with them.
BETTY is the component that designs and generates the output in the animation mode. A global presentation planner selects relevant information, part of which is allocated to BETTY whenever it should be conveyed via animated presentations.

AVI 94- 6/94 Bari Italy

The content realization is done by the animation component including scripting, camera motions, cutting, computation of the single frames and playback at presentation time.
In the following, I would like to restrict the term "animation" which initially includes every kind of animated graphics on a screen to the special case of animated graphics of threedimensional objects.

Why Animation?

The presentation of information as an animation has two obvious advantages: One is that it is possible to show the dimension "time" immediately, for example a duration or succession. Temporal events can be presented without metagraphics directly as they happen in reality. Using a text, it is also possible to express the dimension time in a certain way, but the clearness and shortness of a visual presentation is far more direct than the description of some temporal information by natural language. In static graphics temporal statements can only be made by metagraphical elements such as numbers or iconized clocks or in the form of picture sequences.
The second advantage is, that in an animation it is much easier to (virtually) cross the border to the third spatial dimension (which the screen naturally can't really show). Static graphics can only use the means of perspective and projection for this purpose.
The human visual system produces the threedimensional imagination of objects in the world by viewing it from different positions. Now a static graph can only show one aspect of an object in contrast to an animation, in which the camera can be moved around the object. Changing the position of the camera makes 3D objects become plastic. This allows the visualization of spatial relations of objects to one another in the 3D world precluding a single 2D projection's or a natural language presentation's detour and loss of information.
Finally, movements of objects are nowhere else as easy to show as in animated graphics, so literally, the best way to show a motion in a picture seems to be a motion picture...
All these are good reasons to further investigate the automatic generation of animations, to attempt to solve the tasks of scripting, filming, cutting and to explore the se-

mantics of the elements of filmmaking. The recent work done by [BW90, KF90] also shows the growing interest in this topic.

The BETTY System

When a multimodal document is planned by the WIP system, a superordinate presentation planning component selects the information to be communicated and allocates it to the respective design and realization components. The results of these mode-specific generators are then coordinated by a layout component on the output medium such as a screen or printer (see Fig. 1).

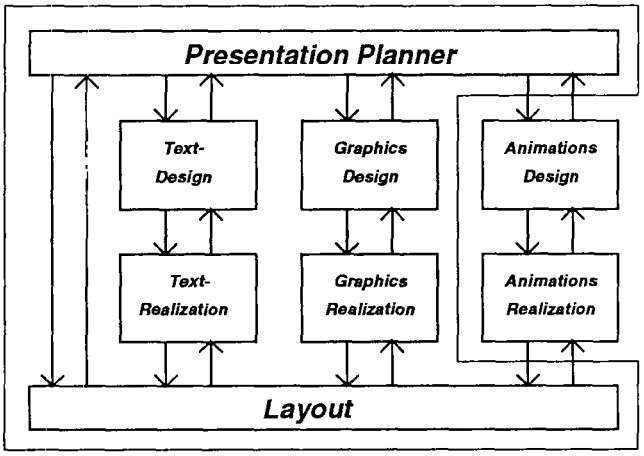


Fig. 1: Simplified architecture of the WIP System

The problem of designing an animation can be considered as a planning problem and the input of the animation design component amounts to visualization goals that specify the contents to be transmitted in the sequence, as well as some generation parameters. Such visualization goals are for example the localization of a certain part of a device. The animation sequence then has the task of showing the user where this part is located. Other goals are the demonstration of movements of objects or the visualization of the construction of assemblies.
This latter one can be done by animating an explosion of the complex object showing the single parts moving away from each other and allowing the user to see objects that were previously hidden.
On the output side of the design component the whole animation has to be described in such a way that all the object and camera positions and motions are fixed at every time in the sequence. This script, containing the complete temporal and geometric information, can easily be realized by a given animation system, such as Soft-Image or S-Dynamics.
The knowledge that is needed for this design process is extracted in different ways: The wire frame models of the objects to be animated are stored in a CAD System, which implies that they can be accessed unconditionally in computer aided construction processes. The knowledge of logical structure of the devices like the part-of hierarchy or the "side information" (which side is the front side?) is stored in symbolic form and it is also used for the generation of language and static graphics.
The representation of the object movements is the same as that used in WIP for the design of static graphics depicting movements. Arrows, ghost images and animation sequences describing the same movement are generated from the same internal representation (see also [SK93]).
Finally, the knowledge of filmmaking is encoded in the decomposition rules of the planning component and is completely independent from the objects or domains. The formulation of these rules was mainly based on common design principles ([Ma75]), the attentive study of TV and cinema films and the stepwise experimental refinement. In addition, [KF90] state some principles and fundamental issues for camera positions and movements, that serve as a good guideline.

The planning process

BETTY's design component plans a script for an animation that satisfies the visualization goals submitted to it by expressing the contents specified by the presentation planner. The specification of such a script can be done at different levels of abstraction, wherein the abstraction hierarchy stated in [Zl90] yields a good starting point:
The uppermost layer is the description of the task of an animation (task level), which is described by the visualization goal. This task is performed by applying various filmmaking techniques that fulfil certain purposes and functions (functional level). The techniques themselves are realized by certain basic procedures such as moves, zooms or cuts, which build the procedural level. Finally, on the bottom end, all movements of camera and objects have to be described explicitly to the machine (machine level) which is a sad reality for human animation designers with many commercial animation systems.
From this abstraction hierarchy, it becomes obvious that the whole script can be described in a hierarchical manner as a tree or any other nested structure. A sequence consists of several subsequences which can be parallel or sequential in time. As an example, a sequence showing a movement of an object always has to consist of at least two parallel subsequences, one describing the camera positions, and one describing the object motion. The leaves of this subsequence-tree are basic movements of objects or the camera at certain times along certain axes or trajectories.
From this data structure we can immediately derive an algorithm to create a script: the top-level visualization goal (task level) can be decomposed into several subgoals by decomposition rules. These subgoals (functional and procedural level), either parallel or sequential in time, are decomposed again by other rules until there are only elementary actions left. These actions can be translated into elementary movements at the machine level, which build up the whole script when we go back in the tree. An example of this kind of decomposition rule (in a LISP-oriented Syntax) is the following:

```
(anim-rule move-view-to-object (part time)
   :PARALLEL
   (move-cam-to-object   part time)
   (move-aim-to-object   part time)
   (keep-view-angle      time))
```

This simple algorithm already serves as a good starting point, but often leads to redundant scenes and useless motions. The same goal can imply different subgoals in different situations that we can see from a simple example: If we want a zoom into a part on the front side of a device and the camera finds itself at the back side, there has to be a cut or even better, a move around the object just to make the part visible in the viewport before the zoom can be performed. On the other hand, when the camera is already in front of the object, another move or cut would be useless or even totally invisible, resulting in a waste of time in moving from point A to A.
So, the system has to maintain some concept of its actual state, i.e. the position of the camera and the involved objects not only in terms of absolute coordinates, but also in symbolic form. Such a symbolic context allows the planner to split a certain goal into different subgoals in different situations and what we only have to do is specify the decomposition rules in a suitable manner in order to adapt the results to the single situations. A conditional decomposition is specified as follows:

```
(anim-rule show-object-motion (part motion time)
   :COND
   ((and  (camera cam-pos part)
          (camera aim-pos part)
          (camera viewangle-for part))
    (show-motion-cond-1 part motion time))
   ((and  (camera cam-pos part)
          (camera aim-pos part))
    (show-motion-cond-2 part motion time))
   (t
    (show-motion-cond-3 part motion time)))
```

Depending on certain conditions of context (like camera position, aimpoint or viewangle), one subsequence is then selected in the example above.

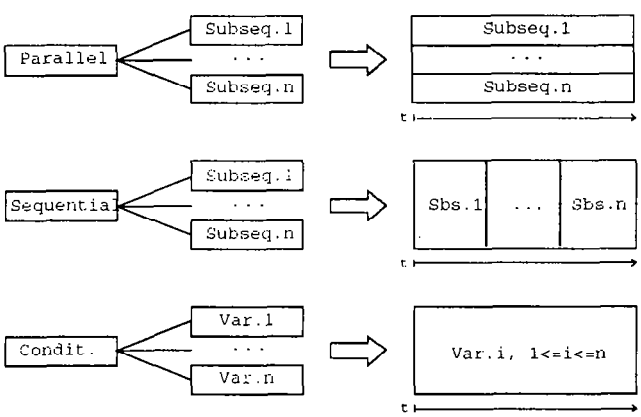
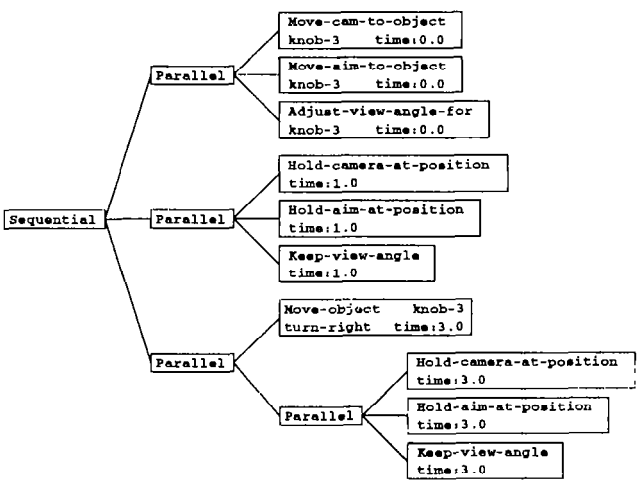
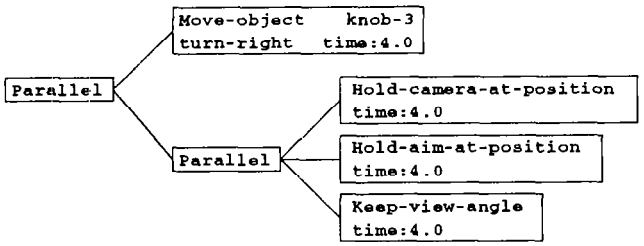


Fig.2: Three kinds of decomposition and their effects in the script

The decomposition of a goal into several subgoals can be specified by a decomposition rule as either parallel or sequential in time or as a conditional decomposition that selects one subgoal out of several depending on the actual context. These three concepts and their effects in the planned script are visualized again in Fig. 2.
At the bottom end of the hierarchy, the planner finally has to specify exact camera positions in terms of coordinates in the CAD system. These computations are done by a component for the choice of perspectives that is also used for the generation of static graphics and which is described in more detail in [AR90]. The motions that constitute the explosion of an object group are computed by the same component that creates the explosions in static graphics in WIP (see [SK91]). By using these components for the choice of perspectives and the computation of trajectories, that are already used in other parts of the same system, the overall amount of code and data is largely reduced resulting in a better system performance and shorter development times as well as in visually consistent presentations.
Furthermore, the absolute object and camera positions are held up to date in the CAD System during the planning process, so that there is always a link between the symbolic represented situation and the absolute coordinates. With this procedure the same visualization goal can be transformed into completely different animations by the same set of decomposition rules depending on the actual contexts in which the planning takes place. A fitting example is presented in the two following figures, that show the tree structures of different decompositions of the same subgoal in different contexts. In both cases the task of the animation is to show within 4 seconds how to turn right knob-3.
In the first situation, the camera is at the wrong side of the scene, so that there has to be a cut (i.e. a move in time 0) to a suitable position before the movement of knob-3 can be shown. After the cut a short still shot is inserted to avoid confusion about the new camera position.



In the second situation, the camera is already in a fitting position, so that the demonstration of the movement can immediately take place.



Another important feature of this method is the following: If we want one animation to fulfil several visualization goals subsequently, then these goals don't have to be all known at the beginning. The system appends a new sequence to an existing animation considering the context resulting from the satisfaction of the earlier visualization goals. Thereby we get a single animation including several actions or statements whose flow is not disturbed or interrupted. Furthermore, this method makes the system work incrementally, which means that planning and presentation time can overlap. This is especially necessary if we want a close-to-real-time document generation with the ability to react to feedback from the viewer (see also [PPP]).

Like in the ESPLANADE System described in [KF93] the planning approach taken here is a hierarchical one, but taking a closer look, there are important differences between the two systems. While ESPLANADE first decomposes an animation into several sequences, then all these sequences into scenes and finally all scenes into shots (breadth-first), the BETTY System works depth-first. The advantage of this approach is that all the details from earlier shots are known when a new shot is planned, which allows the system a strong lookback, but no lookahead (in contrast to ESPLANADE where there is both lookahead and lookback, but only to the actual level of detail (and the levels above).

The Realization

To make the system adaptable to different animation systems, we have developed an application-independent description language for scripts at the output side of the design component. Descriptions in this language, that include all low-level information as well, can be translated by different backends to the specific data structures of the respective animation system. These backends also provide some means to extract geometrical data from the animation system and to control the computation and the playback of single frames.

This has the advantage that the design process is independent from the specific output program, as long as this program at least follows the camera-paradigm. Since the backend strictly separates the design and the realization components, these processes can even run on different machines allowing us to use specialized hardware for the sake of higher performance.

The animation planner itself is even able to work in real time in a certain sense because the script is generated in an incremental way and the computation times are very moderate. What takes time is the computation of the images, rendering, raytracing and so on, but, if one is willing to accept simple wire-frame output, the component can work very closely to real time.

This opens the perspective to use it in other domains such as technical control units where the presentation of complex data in compact form is a difficult task. The overall architecture of the BETTY System is shown again in Fig.3:

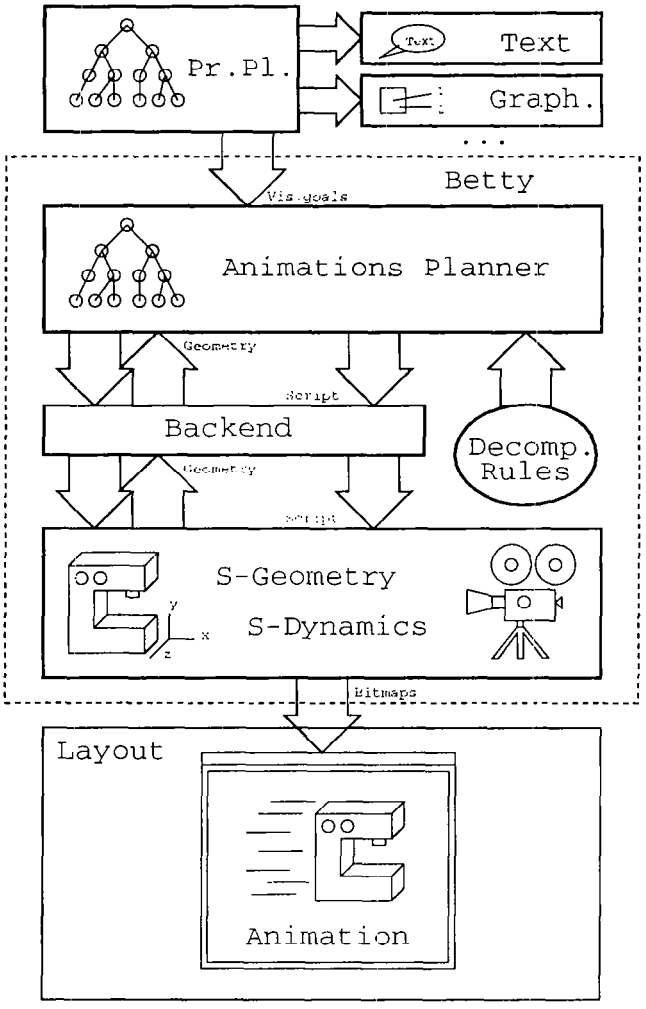


Fig. 3: Architecture of the BETTY System

The Demonstration Environment

For the documentation and testing of the system an interactive presentation environment exists, although the component itself is not intended to work interactively (not to confuse with the fact that it produces presentations that require an interactive medium...). In this testbed, it is possible to specify visualization goals via menus and to immediately see the results of the planning process as well, in the form of tree structures as in the form of ready-made animations. The effects of changing the set of decomposition rules can be investigated immediately and the refinement of these rules (by trial and error) is strongly simplified.

Figures 4 and 5 show snapshots from two animations generated by the system. In Figure 4 the input visualization goal is to show the movement of taking away the cup from an espresso machine. The stationary camera zooms in on the cup which is placed on the machine table, the cup starts to move, then, after a while the camera sways a little to center the cup in the image and then escorts it as it moves further away.

So, through this arrangement, we managed to see that only the cup is moving, not the machine (train window effect); and by escorting the cup at a later stage we didn't lose it out of the picture, even though we were close up.

In Figure 5 there are two input visualization goals: the first one is to explode the front right wheel group of the lawnmower, while the second goal is the locaization of a detail in the wheel group.

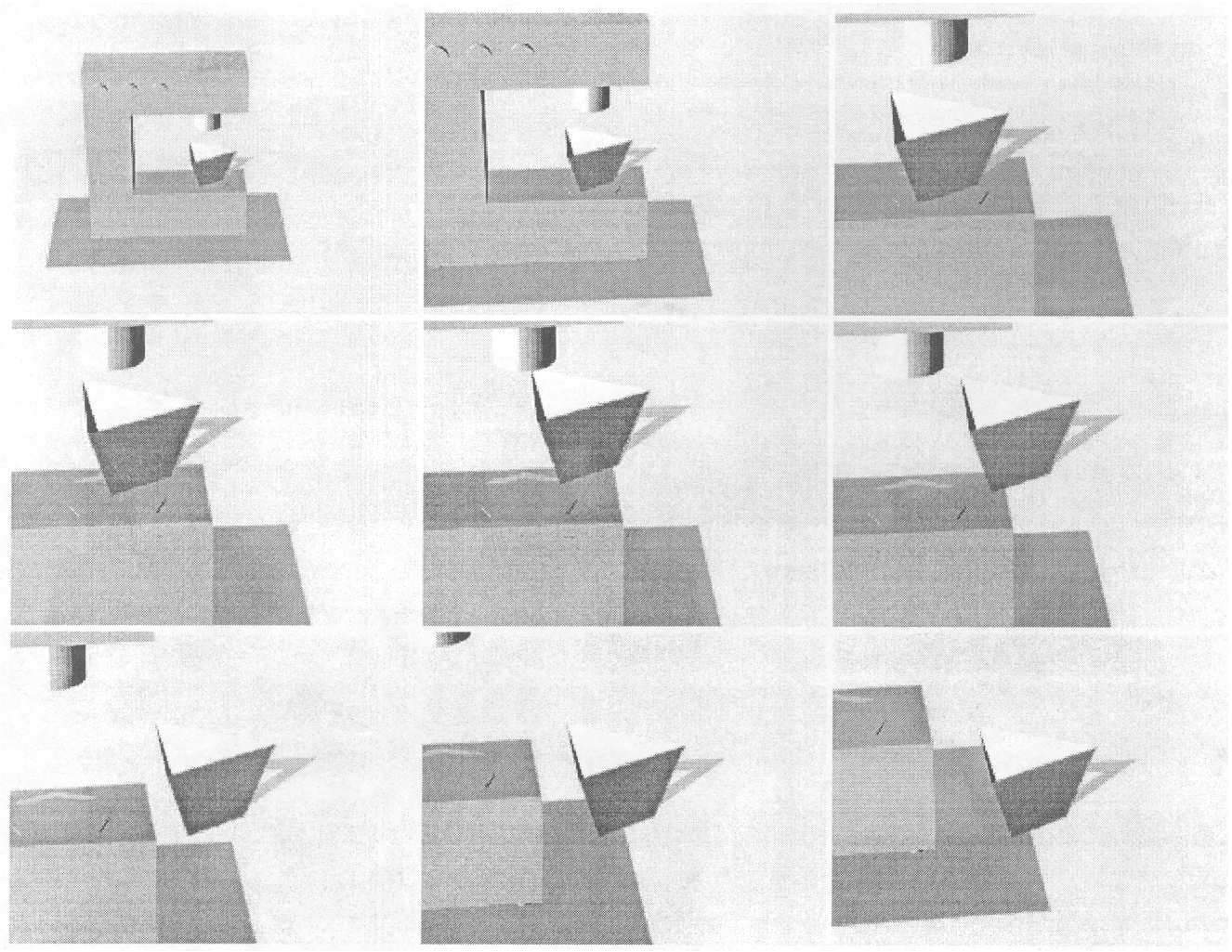


Fig. 4: Snapshots from an animation showing an object movement

Conclusions and Future Work

The approach taken in the BETTY system leads to expressive informative animation sequences that clearly communicate the intended information. With a look to interactive multimodal presentations ([PPP]) the planning component was designed to work closely to real time. Furthermore, the generation speed of the whole animation component depends mainly on the image computations of the respective graphics system.

The animations generated so far, only include the 3D models of the involved objects, but as with static graphics one can also think of the use of metagraphical elements in animations. For example a symbolized hand could move a part instead of leting it move by itself. One can even contemplate annotations that appear in the running animation. Another interesting feature would be to animate actions that have no direct geometrical appearance in the 3D model, such as heat or sound.

They could be symbolized again by metagraphical elements like symbolized sound waves or changing colours of a part: A tube that gets hot could turn red, a cold one blue and so on. All these ideas have to be refined and tested in the future. Additionally a more sophisticated approach to the choice of light settings is necessary for the generation of rendered 3D animations. The choice and guidance of light sources seems to be at least as important in the

generation of these animations as the choice of camera positions and settings.
But all these features still seem to fit into the existing concept of the BETTY system since the decomposition rules describing camera motions can include descriptions of light positions as well. The motions of metagraphical 3D objects such as hands or sound waves can be described in the same way as the motions of the "main actors", and the change of the colour of an object is from a conceptual point of view not much different from the change of its position.
Finally, the layout problems in the temporal dimension have to be dealt with and the animation component has to prove its power and usefulness in practical application.

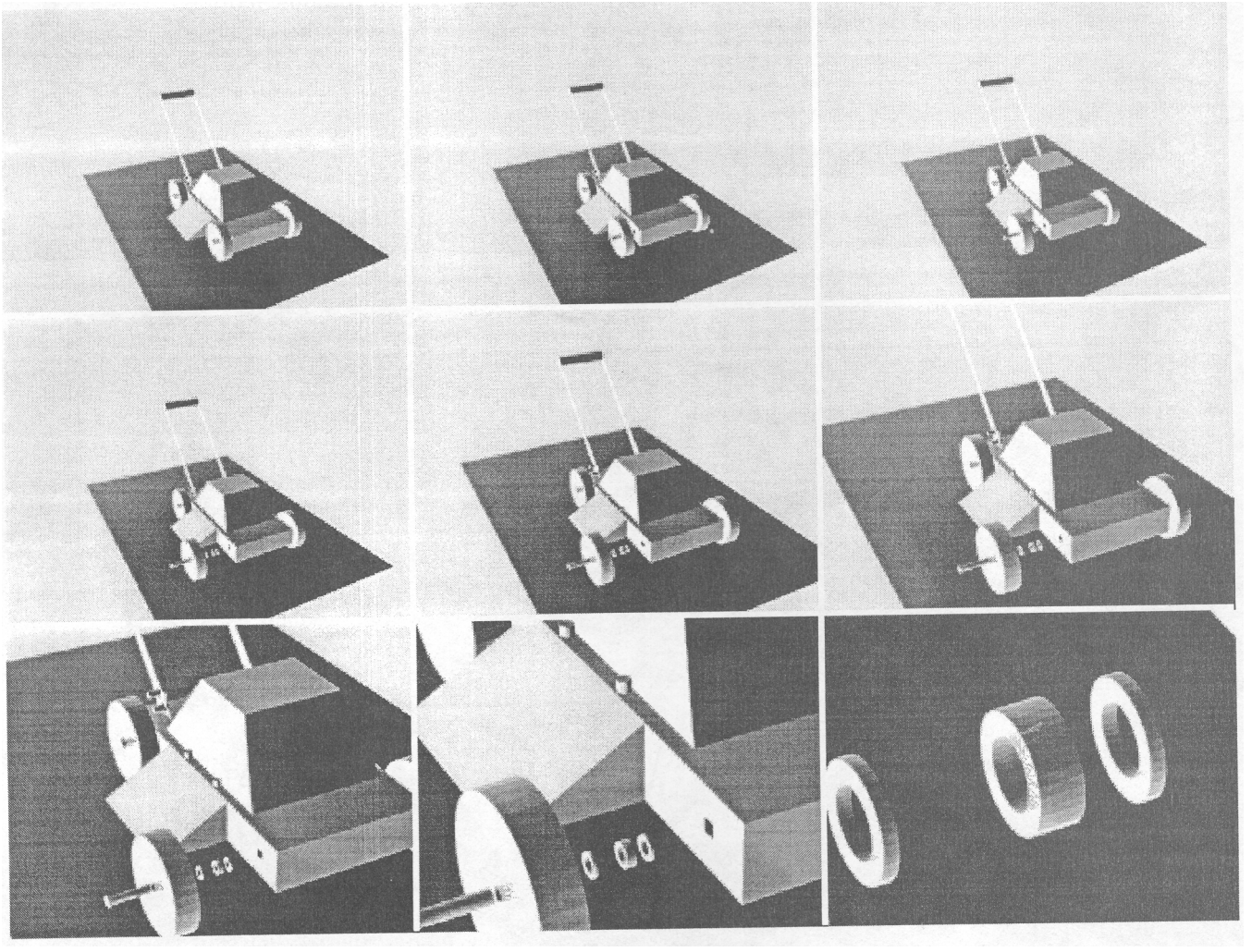


Fig. 5: Explosion of an assembly and Localization of a part

Literature

[AR90] Elisabeth Andre, Thomas Rist, "Wissensbasierte Informationspräsentation: Zwei Beiträge zum GI-Fachgespräch Graphik und KI" In: Research Report RR-90-07,1993 DFKI GmbH, Stuhlsatzenhausweg 3, 66123 Saarbrücken.

[BW90] Norman I.Badler, Bonnie L. Webber, Jugal Kalita, Jeffrey Esakov, "Animation from Instructions" in Norman Badler, Brian Barsky, David Zeltzer: "Making Them Move: Mechanics, Control and Animation of Articulated Figures" 1990: Morgan Kaufmann, 50-93

[Fn89] Steven K. Feiner, "Specifying Composite Illustrations with Communicative Goals" Proc. UIST '89 (ACM SIGGRAPH Symposium on User Interface Software and Technology) Williamsburg VA, Nov 13-15, 1989, 1-9

[Ma75] Jörg Michael Matthaei: "Grundfragen des Grafik-Design", Heinz Moos Verlag München, 1975, ISBN 3-7879-0081-0

[KF90] Peter Karp, Steven Feiner, "Issues in the Automated Generation of Animated Presentations", Proc. Graphics Interface '90, Halifax, 14.-18.5. 1990, 39-48