

SECTION 5

AMI RESEARCH ROADMAP

5.1 CRITICAL TECHNOLOGIES

Two broad technology areas are critical to the design and development of a TO validation tool like the one described in this report. These areas encompass both software components and modeling framework. In general, the software components correspond to the simulators, software agent, rendering system, user interface, translators, and data manager. The modeling framework defines the standards that must be followed to engineer correct and reusable models for each domain.

The user interface and PDM components are mainstream technologies. Rendering and scene management systems are also reaching maturity. Although some are stand alone, others work as an integrated environment where various data and behavior sources interact uniformly. Human models are also available off the shelf. So far, we have been using the Jack Toolkit as an implementation candidate; however, other models exist. In any case, it would be prudent that the design of a TO validation tool be compatible with emerging industry standards for human models and 3D geometry [14].

5.1.1 Human Models

Most human models today possess the basic capabilities needed to execute a task; they can reach and look, and walk and pose. Most models can change shape and size to reflect variations in human anthropometry; some even can adopt a specific person's body shape taken from laser or video scanning. The better models can be animated via procedural codes, motion capture, or interactive manipulation. The best models can be controlled through program interfaces and enjoy high-level behaviors such as attention, coordinated full body reach, balance, and collision detection. A desirable feature, not yet found in commercial human modeling systems, would be a direct linkage between strength, fatigue, comfort, collision avoidance, and task achievement. Inverse kinematic procedures can manage collision avoidance and task achievement, while dynamics simulation possesses all five features. True dynamics simulations are both expensive to simulate and difficult to control, and are not likely to be readily available outside the research or other specialized

communities (such as sport performance analysis or clinical biomechanic studies). However, most tasks in the aircraft maintenance domain are not characterized by fast, forceful movements – more likely by awkward postures, torque strength, repetitive actions, or hazardous substances. None of these situations requires true force-based dynamic simulations, so inverse kinematic procedures will usually suffice.

In order to function within the TO validation domain, a human model should be able to understand and execute tasks or procedure steps, preferably stated in a form convenient to the user. The software structure of the human model should facilitate access through a well-defined functional API and should permit the return of model state information useful for evaluation and validation. Ideally, such information will be used to guide or modify the simulation, thereby providing some task responsiveness in lieu of actual force-based (dynamic) simulation. For example, a reach task failure may trigger alternative access paths, collision detection may be replaced by collision avoidance, and an occluded line of sight may cause automatic re-posing of the human model. These are precisely the situations appropriate for task validation: feasibility is more important than optimality. No existing human model meets these requirements, but one with a good API and reporting facilities will be clearly superior. An instruction-level control and simulation system will fit comfortably on top of such an API. We next turn to examine a representation that will allow instructions to the human model (with a suitable API). With an instruction-level interface, the TO author should be able to launch human action validation studies from the TO text, see the results of the validation in computer graphics, and examine any resulting failure conditions.

5.1.2 PAR-Based Agent and Specific Skills

The AMI framework relies on the use of PAR scripts to model maintenance procedure subtasks and an Actionary to model the knowledge of the technician. We still have to prove that PAR is suitable for large-scale Actionaries. Furthermore, skill-specific technologies need to be integrated into the PAR system. In particular, (dis)assembly planning is a “must have” in the domain of maintenance simulation.

Because of its generality, PAR is not expressive enough to capture the complexity of a (dis)assembly operation in a flexible way. For example, one could script a whole disassembly sequence as in the editing process. However, writing such a script would be

labor intensive for large assemblies. In particular, the author must select an assembly sequence that guarantees that the assembly is stable at all times. This is known as the fixturing problem or finding areas to support or grasp an assembly to counteract its weight and insertion forces. In the uplock hook scenario described earlier, the author must instruct the technician to hold the hook with the left hand.

General assembly planning problems are currently too complex to be solved by computers. State-of-the-art assembly planning algorithms [15] will only handle simple problems. Simplifying assumptions such as one-step motion, one-step translation, and monotonic sequences means that the insertion path for each part is defined by a single rotation/translation or a single translation. Furthermore, the sequence cannot undo or temporarily reconfigure a subassembly to enable other insertions. Assembly planning research also addresses related problems such as fixturing [16] and use of assembly tools [17]. Few robust assembly planners exist. The most widely recognized as such is Archimedes 2 [12].

Using an assembly-planning module extends PAR functionality, but more importantly, it allows the author to let the virtual technician solve assembly problems as a human technician would and only detail critical tasks. The assembly skill allows textual and scripted orders to remain at the same level of abstraction. An assembly-planning module would allow an author to script assembly related orders the way they are naturally issued; i.e., with goals and constraints rather than with detailed assembly steps.

The uplock hook example shows that in spite of the disassembly skill of the technician special constraints have to be made to explicitly prevent hazards. In particular, the third TO simulation instructs Jack to unscrew the bolts without removing them from the hook. Assembly planners are also meant to handle such constraints [18].

5.1.3 Natural Language Technologies

In previous Air Force projects addressing TO generation, our research group investigated issues involving natural language understanding and generation. The theory behind this was based on the fact that TOs are written in natural language, not an artificial or algorithmic one. Therefore, the TO authoring process had something to do with the creation of such natural language text. As we studied the problem further and consulted TO authors,

the role of natural language shifted from generation more toward understanding. The main reason for the shift was that existing instruction sources -- either as TOs or as LSA records -- could be a resource in building the procedural step representation, or PARs. Once the maintenance task was described in PAR form, it could be edited, animated, and used for task validation. Moreover, the PAR form by design lends itself to natural language sentence generation should that be necessary or required.

Natural language technology can be used for TO validation under the following conditions:

- A natural language parser must understand the syntax of the sentences it is presented.
- A natural language parser must have a lexicon so that it can understand the words used in the instructions. The technology we use for this involves a particular kind of parser that uses tagged fields for each word (so-called lexical semantics) to properly interpret the input sentence.
- The parser must output its sentence analysis in a form that is digestible by other processes; in particular, we demand that the output be in an action representation form (PAR) suitable for subsequent control and animation of a human model.
- The natural language processing from sentence to PAR should occur fast enough to be transparent to the user of this technology.
- Natural language processing should eventually be satisfied by commercial off the shelf (COTS) components.

In the PAR implementation that we have developed, natural language technology is used to build the proposed framework to validate TOs. Our software module takes natural language instructions and generates one or more instantiated PARs. The basic linguistic representation of an action is a predicate-argument structure such as 'slide(John, box),' which indicates a particular action (the predicate 'slide') and its participants (the arguments 'John' and 'box'). We use the XTAG Synchronous Tree Adjoining Grammar System, which consists of a parser for extracting the predicate-argument structure of an input sentence, and a translator for generating an instruction script from this predicate-argument structure. The parser extracts these structures by first associating each word in an input sentence with one or

more elementary tree fragments, which are combined into a single derivation tree for the entire input sentence using the constrained operations of the XTAG Synchronous Tree Adjoining Grammar System formalism. These elementary tree fragments have argument positions for the subjects and objects of verbs, adjectives, and other predicates, which constrain the way the fragments can be combined, and which determine the predicate-argument structure of the input sentence. The translator then converts this predicate-argument structure into an instruction script, which in turn generates one or more instantiated PARs. With this architecture, a wide variety of inflections and grammatical transformations can be reduced to a much smaller set of predicates in the parser, and a variety of synonymous predicates can be further reduced to a still smaller set of PARs and scripting-language keywords in the translator. Although some parts of the translator may be domain-specific (some actions may depend on particular objects in a domain), the parser can easily be ported between domains, since its predicates are based on linguistic observations instead of on a particular programming language or virtual environment.

5.1.4 Semi-Qualitative Simulation

Semi-qualitative simulation has mostly been applied to build virtual laboratories. It has not yet been used for large-scale applications. We are currently developing a new semi-qualitative modeling language with standard object-oriented features that should help create and maintain large model libraries [19].

We are also addressing performance issues to reduce the lag between quantitative and semi-qualitative simulators. This difference is mainly due to the ability to change the structure of the simulated system during a simulation. This feature is required for specific applications such as maintenance simulation.

5.2 OUTLINE OF FUTURE TASK EFFORTS - FY2000

5.2.1 Represent Procedure Steps with PAR

Continue research to represent procedure steps with the Parameterized Action Representation (PAR) to describe how language inputs can effectively create PARs for downstream simulation and validation. The PAR allows a media-neutral form in which task instructions and their execution requirements may be stored for later retrieval, re-use, and simulation. By establishing a correspondence between the PAR parameters and the objects

and situations being examined, the PAR actions can animate a human form maintainer model such as Jack. The effort should focus on the feasibility of creating PAR instances from language and instruction analysis sources such as LSAR records, existing TOs, and the author's conception of the task.

Four tasks comprise this two-year effort: (a) create PARs for selected maintenance tasks, (b) investigate the requirements to correctly parse LSA records and produce or select PARs for them, (c) determine how to convert spatialized descriptions (in LSAR) to draw references in TOs, and (d) collect TO author monologues during changes and updates.

5.2.2 Validate TOs through Automatic Generation of Virtual Motion Simulation

Demonstrate TO validation by automatically generating virtual human motion simulations. This will consist of simulated assembly and disassembly tasks based on TO procedure steps, and should consider validation-critical issues such as confined reach task planning, spatial reasoning for part and assembly removal and replacement, and qualitative modeling of object function and behavior during maintenance tasks.

5.2.3 Determine Knowledge Representation Requirements

Determine the necessary knowledge representation requirements to actually deploy automated maintenance instructions. Beyond demonstration systems, there are real and significant issues related to obtaining and managing the large amounts of data, part information, CAD files, and the engineering schematics necessary for TO generation and validation tool. The requirements for a usable and scalable system need to be outlined.

This two-year program would include five tasks: (a) demonstrate that PARs for selected maintainer tasks can be simulated on a human model; (b) detect and report PAR simulation failures; (c) design software interfaces so that motion optimizations can be used if needed, but are not called if feasibility is more easily shown; (d) survey and establish priorities for human task functions that may need to be simulated; and (e) determine if the task analysis components of human models can be actively used during simulation to check task feasibility.

5.2.4 Create PARs Through Human Performance Motion Capture and Semantic Analysis

Use human performance motion capture and the semantic analysis of those motions to construct PAR patterns (called UPARs: uninstantiated PARs) for typical maintenance activities. Human motion collected in a VR environment may be used to represent either coarse or fine motion strategies for part removal and replacement. Investigate how VR inputs and outputs impact the generation and use of PARs for maintenance actions. Since PAR is a media-neutral form used for action representations, the outputs that may be obtained from PARs should also be media-neutral. Investigate such media-neutral representations, for example, XML for multimedia markup and interpretation. Develop demonstrations that show how PARs can use a media-neutral output representation and how they may be variously interpreted in textual or graphic fashion.

5.3 BRIEF OUTLINE OF FUTURE TASK EFFORTS - FY2001

Extend the task validation via human form and system simulation. Candidate extension capabilities could be enhancements to the geometry/function reasoning system, improved performance in complex geometric situations, visualization of human interaction (contacts, pressure) with objects, and automatic annotation of maintenance-significant part features. The system should also provide reports on the cause of any validation failures. Such information would be used to inform the TO author of possible flaws in the task procedures. In the future, such information may be provided to automatic procedure planners who may attempt to reformulate the procedure steps or recommend other geometric alternatives to the concurrent design team.

Investigate the use of natural language as a direct means of modifying existing task PARs. This will be done initially using a fixed-initiative mode of interaction with the computer initiating the dialogue. The investigation should be expanded in the future to allow for more natural interaction, with more user initiation, and greater range of input modalities such as gesture.

Create context filters for multimedia presentations of TOs. Context is used to establish what information is presented and in what form. Different situations may require the same information be filtered and output differently. Examine the feasibility of presenting

TOs in forms useful to authors, maintainers, instructors, and trainees, including thumbnail stills, animations, and speech. Determine the role and usefulness of XML or other alternatives for these functions.

Demonstrate the prototype system on a typical TO generation, validation, and presentation task. Report on the process and recommend areas for further study as well as those ready for more systematic development.

5.4 DEVELOPMENT PROGRAM

In addition to integrating the results of any previous programs into electronic TO authoring systems, the following issues must be addressed, resolved, and implemented in any future development program:

- The PDM requirements must be defined, preferably with standardized terms and data requirements for maintenance features, object function, contents, etc. It may be best to select a target CAD system and its associated PDM and define the needed framework.
- Access to engineering and simplified CAD data on assembly shape, structure, and part function is needed to assess maintainer hazards, (dis)assembly orders, and equipment limitations. This data may be available through the PDM, but it may be scattered across enterprise databases and non-integrated software systems.
- A human modeling system interface should be based on a human modeling standard, or at least on a standardized API.
- A few robust extensions to human form animation systems need to be developed, especially collision avoidance reach planning and action failure reporting via the API.
- PAR and natural language parser software must be migrated into the TO authoring environment.
- Visual and textual interfaces must be implemented to launch validations and interpret their results within the authoring workstation.
- TO prototypes must be evaluated and iterated with real authors performing actual authoring and update tasks.

Since the issues outlined above are part of a large-scale software effort, a competent software integrator should bear prime responsibility. Software components from the proposed FY2000 efforts need to be incorporated and possibly extended. While COTS components such as human models may be available for certain aspects of the development effort, a CAD visualization tool, a language parser, and the baseline electronic TO authoring system, ongoing dialogues between contractors will clearly lead to increased likelihood of successful integration and product performance.