Title:
Semi-Automatic Task Planning of Virtual Humans in Digital Factory Settings

Authors:
Martin Winter, martin.winter@mathematik.tu-chemnitz.de, Technische Universität Chemnitz
Thomas Kronfeld, thomas.kronfeld@informatik.tu-chemnitz.de, Technische Universität Chemnitz
Guido Brunnett, guido.brunnett@informatik.tu-chemnitz.de, Technische Universität Chemnitz
Helge Ü. Dinkelbach, helge-uelo.dinkelbach@informatik.tu-chemnitz.de, Technische Universität Chemnitz

Keywords:
Virtual Humans, Digital Factory, Human Factors, Assembly Design, Process Planning, Virtual Prototyping, State Space Search

DOI: 10.14733/cadconfP.2018.283-287

Introduction:
Digital factory planning and digital commissioning are important tools for process and production planning. Their main objective is to minimize costs by optimizing the factory layout in terms of process time and material flow. The focus of these planning methods recently has changed from the simulation of machine and material movements to a human factor oriented viewpoint – especially considering an aging workforce. In this process ergonomic methods are used for the evaluation and the identification of potential problem areas. At the moment, production simulations involving digital humans are mainly used in major enterprises, operating in the automotive and engineering sector. The high operational costs of available software systems, e.g. for necessary staff training, in combination with the enormous expenditure of time for creating a simulation, are substantial shortcomings which make this approach nearly inaccessible for small and medium-sized companies [9] [5]. To improve the usability of digital humans in simulations, methods are needed that avoid the time consuming process of manual programming by automatically creating work plans, actions and movements for digital humans. In this paper we present a software system "The Smart Virtual Worker" (SVW) [6] that has been developed as an experimental platform to demonstrate the feasibility of such an approach.

The SVW is capable of performing five different types of elementary actions (walkto, transport, align, assemble and disassemble) and for any particular action that is performed our software calculates ergonometric scores based on Method Time Measurement (MTM, [3]) and Rapid Upper Limb Assessment (RULA, [4]) for their evaluation. An Autodesk® Inventor® plugin [1] to our system is used to create a realistic virtual environment for the SVW. A task description for the SVW can either be imported with this plugin or created with the GUI of our software. Tasks for the SVW are internally represented as objectives. This means that any task is defined by criteria that must be met (the terminal objectives), rather than a specific action sequence to reach it. The actions to achieve the terminal objectives are generated by the planning algorithm that intends to minimize an objective function specified as a convex combination of the MTM and RULA scores of the movements of the virtual worker. The user can influence the optimization by providing the weights to the convex combination.

The task of finding optimal action sequences can be modeled as a shortest path problem in an abstract state space. However, the state space is fairly large and unstructured (to keep generality and extensibility)

and we can only compute costs and states as we traverse the space. Therefore, a world agnostic learning method seems appropriate, more precisely a reinforcement learning (RL) based state space search is used to determine an ergonomically optimal solution to the planning problem. See [7] for a detailed description of general RL techniques, and [8] for an exemplary application of RL methods in the SVW.

The RL agent has no a priori knowledge about the world and consequently does not distinguish between reasonable and unreasonable actions in any particular situation. Therefore the agent creates a tremendous amount of useless action sequences (until enough knowledge has been build up in the reward mechanism of the learning strategy) which results in high expenses for run-time and memory consumption. In this paper we report on our approach to reduce the computational costs by combining the RL with a reasoning mechanism, which pre-selects reasonable actions, and therewith counteracts the combinatorial explosion of the search space. To formalize the selection mechanism we introduce the concepts of objective spaces and their predecessors. An objective space O represents those states of the environment that satisfy given objectives and a predecessor is the collection of all states which can be transformed into elements of O with a single action of the SVW. By recursively generating the predecessors of objective spaces and storing them in the nodes of the objective tree we create a data structure that represents preconditions for actions to be performed. A locking mechanism on this tree is used to specify objectives that have to stay valid in the course of actions. Actions are considered invalid, if the achieved states do not satisfy all locked objectives. In this way, we can avoid undoing achieved goals, and hence, avoid getting stuck in loops. By systematically locking/unlocking objectives, it is possible to narrow down the set of possible actions, and therewith avoid the combinatorial explosion on the state space search.

In each cycle of the planning process the elements of the action sequence are generated one after another. Each action is determined in a process consisting of three consecutive steps: selection of achievable objectives, action pre-selection and action selection. Achievable objectives can be fulfilled with a single action in the current state of the environment. For each achievable objective the action pre-selection determines the corresponding actions and sorts out unreasonable actions. This list of actions is then handed to the action selection based on reinforcement learning. It decides on a single action from the list, and sends it to the environment for application.

Factory layout and product setup:

As an important step of the virtual prototyping of factories this layout is a significant aspect of the SVW framework. The factory layout contains instances of objects from the asset library. Each asset includes different variants of an object plus several properties to customize the object. The specified instance has a defined landing surface. This ground plane is used to place the instance on the floor or on top of other instances. Furthermore, one can define different types of connectors, which allow the alignment of two objects based on the axes of those connectors. They can be created e.g. from a planar surface, a hole center or a vertex.

The product to be made is setup as an assembly model – a collection of parts and sub-assemblies that function as a single unit. They are connected by assembly relationships, which control the component placement and the degree of freedom of the connection. To use a product, the product itself as well as its parts have to be contained in the asset library.

Overview of the framework:

The developed system consists of a series of modules that communicate with each other via a middleware. An overview is shown in Fig.1. The GUI module is not included; it interacts with all the other modules for the purpose of configuration and visual feedback.

The central component is the environment module. It manages the scene graph, as well as the current and previous states, and the static and dynamic parameters of the virtual worker. Thus it forms the data

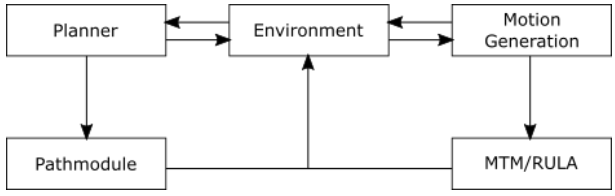

Fig. 1: Scheme of the SVW framework. An arrow indicates the direction of a request.

Fig. 2: Screenshot of the graphical user interface.

base for the operations of all other modules. In addition, action validation methods have been integrated which calculate all permissible actions for a given state (see "Task planning algorithm").

The planning module calculates the necessary elementary actions (e.g.: go, carry, assemble, etc.) for the given complex tasks. For this purpose, an optimization algorithm is used to optimize the order of elementary actions based on predefined parameters. The path planning module calculates the shortest collision free path between the virtual human and a given target position through the current scene. Based on the anthropometric parameters of the worker, the path module is able to determine the accessibility of objects.

One of the more complex steps in the simulation of virtual human models is the generation of realistic and collision-free motion. Therefore, a method was developed which generates human motion based on parameterized motion spaces [2]. The motion spaces correspond to the elementary actions defined in the framework. The planning module generates the list of actions along with the corresponding parameters.

The MTM method was integrated for time standardization and evaluation of the work process. Furthermore, an ergonomic parameter is calculated for each pose based on the RULA method. The calculated simulation is visualized using keyframe animation (Fig.2).

Deduction of the work tasks:

The software "Autodesk® Inventor®" with its extension "Autodesk® Factory Design Utilities" offers the possibility of designing the product to be manufactured and the layout of the associated factory. Based on this data, our system derives the complex work task to assemble the product. In order to calculate a simulation within the SVW framework, one has to transfer the object definitions, their instances within the initial factory layout, and the process to manufacture the final product.

Definition of the objects

An object is defined by four mandatory attributes: id, name, group, and path to the geometry file. An object's description is generated according to the assembly library and the instanced variants. In addition, attributes such as weight, bounding box and contact points can be specified. Contact points are specific positions on the surface of the object on which it can be grabbed or assembled. The grab points must also be specified in Autodesk Inventor, while the assemble points are derived from the component description. Furthermore, for objects such as tables or workbenches, filing and work surfaces can be defined.

Definition of the scene objects

It contains all the essential information for the representation of the scene: included workers as well as objects. Note the differentiation between object and scene object. A scene object is the instantiation of an object in the scene. It has a unique scene-id and a reference to the object-id. A secene-id of a parent is used to model a scene graph. The location of a scene object is always given in the local coordinate system of the parent scene object.

Definition of the complex task

The task is derived from the assembly file of the product. In this file, components of an assembly are pairwise linked by predefined constraints. With the help of Autodesk Inventor, an exploded view is created. We use this view to calculate the assembly order in a semi-automatic method. The specified constraints are converted into a processing hierarchy. Within an SVW-task, the necessary parameters, such as the definition of specific object settings or target position settings, are defined using attributes. For each task additional conditions can be defined. On one hand, a specific execution sequence of tasks can be defined. On the other hand, the place where the task is performed can be specified.

Task planning algorithm:

The planning problem is formalized as a state space search in an abstract state space S (defined by the possible configurations of the environment) from a pre-defined initial state $i \in S$ to some terminal state $t \in S$. The terminal states $T \subseteq S$ are defined as these states in which all terminal objectives are satisfied.

We give a general overview: the planning process consists of *cycles*, each of which returns a sequence of actions that transforms the initial state into a terminal state corresponding to the tasks (represented by the terminal objectives). Several cycles are performed to improve the quality of the results. In each cycle, the action sequences are generated iteratively, one action at a time. To determine an appropriate next action, each iteration performs the following three steps – objective pre-selection, action pre-selection, and action selection.

The general procedure of a single iteration is as follows: Based on the current state, the *objective pre-selection* compiles a list of currently relevant and achievable sub-goals (*active objectives*). This list is handed to the *action pre-selection*, which for each sub-goal selects actions that can be used to reach it. It is taken care that no actions are chosen that will undo already satisfied objectives. This list of actions is then handed to the action decision, a world agnostic decision algorithm that decides on a single action from the list, and sends it to the environment for application. Above iteration is repeated until a terminal state is reached (i.e. all terminal objectives are satisfied).

The process of pre-filtering actions is effective in largely reducing the size of the explored state space. We demonstrate the generation of an exemplary action sequence.

Conclusions:

The described plugin for the "Smart Virtual Worker" framework provides a user-friendly and intuitive application in the context of virtual prototyping of factories. For given descriptions of transport or assembly tasks the SVW computes action sequences to fulfill these tasks. An optimization procedure is used to find solutions that balance the requirements of efficiency and ergonomics according to the specifications of MTM and RULA. Since these scores can only be computed as we traverse the state space, world agnostic learning methods must be used to compute the solution. In this paper we present an action pre-selection mechanism to speed up the performance of such strategies.

References:

[1] Autodesk, Inc.: https://www.autodesk.com/products/inventor/overview, Accessed: 2017-12-20.

[2] Kronfeld, T.; Fankhänel, J.; Brunnett, G.: Representation of Motion Spaces using Spline Functions and Fourier Series, Proceedings of the MMCS 2012, LNCS (Lecture Notes in Computer Science), 2014, 265-282, https://doi.org/10.1007/978-3-642-54382-1_16.

[3] Maynard, H.B.; Stegemerten, G.J.; Schwab, J.L.: Methods-time measurement, McGraw-Hill, New York, 1948.

[4] McAtamney, L.; Corlett, E. N.: RULA: a survey method for the investigation of work-related upper limb disorder, Applied Ergonomics 24 (2), 1993, 91-99, https://doi.org/10.1016/0003-6870(93)90080-S.

[5] Spanner-Ulmer, B.; Mühlstedt, J.: Digitale Menschmodelle als Werkzeuge virtueller Ergonomie, Industrie-Management 26 (4), 2009, 69-72.

[6] Spitzhirn, M.; Kronfeld, Th.; Müller, N. H.; Truschzinski, M.; Brunnett, G.; Hamker, F. et al.: The Smart Virtual Worker - Digitales Menschmodell für die Simulation industrieller Arbeitsvorgänge. In: Homo Sapiens Digitalis - Virtuelle Ergonomie und digitale Menschmodelle. Bullinger-Hoffmann, A.C., Mühlstedt, J. (Eds.) Springer Berlin Heidelberg, 2016, 385-397, ISBN: 978-3-662-504-598.

[7] Sutton, R.S.; Barto, A.G.: Reinforcement Learning: An Introduction. Cambridge, MA, USA: MIT Press, 1998

[8] Truschzinski, M.; Dinkelbach, H.; Muller, N.; Ohler, P.; Hamker, F.; Protzel, P.: Deducing human emotions by robots: Computing basic non-verbal expressions of performed actions during a work task. In 2014 IEEE International Symposium on Intelligent Control, ISIC 2014, 2014, 1342-1347.

[9] Wischniewski, S.: Digitale Ergonomie 2025. Trends und Strategien zur Gestaltung gebrauchstauglicher Produkte und sicherer, gesunder und wettbewerbsfähiger sozio-technischer Arbeitssysteme, Bundesanstalt für Arbeitsschutz und Arbeitsmedizin, Dortmund, 2013.