

# A General Approach for Modeling Robots

O. Rogalla, K. Pohl, and R. Dillmann

University of Karlsruhe  
Institute for Process Control & Robotics  
Karlsruhe, D 76128, Germany  
rogalla|pohl|dillmann@ira.uka.de

## Abstract

*Modeling manipulators has been an important part in robotic simulations. There are various types of robot systems used in today's robotic research and application, e.g. the six axis industrial robots, humanoid redundant manipulators and 4 finger grippers. Therefore, the model's structure can be very complex, requiring techniques for both modeling and simulating the system. "Traditional" simulation packages handle each robot respectively. This paper will present a formal model for arbitrary manipulators or grippers and implement the theoretical ideas into a real modeling and simulation tool.*

## 1 Introduction

The problem of simulating and modeling robot manipulators has been a challenge for researchers for a long time. Several models for kinematic and dynamic simulation have been proposed to allow more efficient path planning [6], collision avoidance, and work cell construction[5]. Nevertheless, most simulation packages are strongly related to specific robots or at least robot types such as a six axis robot with rotational joints. Although commercial software like RobCAD, Igrip or Anysim do include specific simulation packages for each robot, the prices for these simulation packages are considerably high. Additionally, the simulation's flexibility is very restricted with respect to new robot models. There is a clear need for a more general simulation tool. This need is especially great if several robots are evaluated and compared according to how they solve a specific task. A simulation tool is needed for easily modeling and simulating the robot with its geometric, kinematic and dynamic parameters.

The requirements for such a system can be summarized as follows:

- Basic specification for the parameters of manipulators.
- Easy integration of standard CAD-models describing the geometric parameters.
- Visualization of robot's movements to evaluate its performance.
- Flexible algorithms for kinematic and dynamic simulation (forward/inverse kinematic, acceleration profiles, joint and acceleration restrictions) allowing calculations for several robotic systems.
- Modular approach for a combination of different robots and grippers.
- Simple definitions of mount points, tool center points (tcp), and manipulator bases.
- Interfaces to secure the functionality of the system in various applications.

A system which meets these requirements may be used for solving the following problems:

- A library can be established collecting several gripper and manipulator sources. The robot's specific parameters can be stored and are accessible for clients.
- Changing parts or specific attributes, such as certain arm lengths, can first be tested in the simulation to ease the designing process of new manipulators.
- The user can perform tests with different grippers and robot models before the final system configuration is decided. This reduces costs for changing the design of the real system.

- Algorithms for kinematics and dynamics can be evaluated and compared for further improvements.

Although the advantages for a generic structure are obvious, the results are still unsatisfying. In addition to the combination of different modules specialized for such systems as in Igrip or RobCAD, the mathematical library MATLAB [1] provides a toolbox for robotic manipulators. It uses efficient algorithms to solve kinematic and dynamic problems. Geometric simulation and visualization however are not included. The representation of the robotic systems can only be used and extended by skilled programmers.

Gourdeau developed the free simulation package ROBOOP [4] which supports an object oriented approach with MATLAB-like features, but with better performance. Although the package only provides a programming interface, kinematic and dynamic simulations can be created more easily than with MATLAB. To take advantage of this work, ROBOOP was chosen as an essential component for kinematic and dynamic calculations of the robot library  $K^L_jMt$  (**K**inematic **L**ibrary for simulating **M**anipulators), which will be introduced later.

In this paper, an approach will be presented that allows modeling of complex manipulator structures and behaviors on the basis of theoretical fundamentals. Firstly, a general definition of chains will be given. This is the basis for the final data structure that is used for the simulation. Secondly, the final system will be described which satisfies the requirements mentioned in the beginning of this introduction.

## 2 Background

The concept of modeling manipulators as chains proved to be very helpful in that it provided the storage of data, necessary for the simulation process. Creating such a data structure, a formal definition must first be derived.

Section 3 will introduce the reader to the formal specification of chains developed for robot modeling. In section 4, the practical implementation of the theoretical concepts to a real software structure is displayed.

## 3 Definition of Chain

### Definition 1 Chain Structure

A **chain**  $\mathcal{K}$  is defined by  $\mathcal{K} := (\mathcal{B}, \mathcal{L})$  where

$\mathcal{B} := \{B_1, \dots, B_j\}$  and  $\mathcal{L} := \{\Omega_1, \Omega_2, \dots, \Omega_n\}$ .

Members of  $\mathcal{B}$  are named **base**. They are fixed and attached to each other by definition.

Elements of  $\mathcal{L}$  are called **links**. Every link  $\Omega_i \in \mathcal{L}$  moves along or parallel to a plane. It must be attached to another link  $\Omega_j \in \mathcal{L}$  or to a base  $B_i \in \mathcal{B}$  by a joint. Additionally, every link has to be connected to a base through other links.

### Definition 2 Chain Arithmetic

- $\Lambda$  is an **element** of  $\mathcal{K} = (\mathcal{B}, \mathcal{L})$ , if  $((\Lambda \in \mathcal{B}) \vee (\Lambda \in \mathcal{L})) \Leftrightarrow: \Lambda \in \mathcal{K}$ .
- The **attachment** between  $\Lambda_1, \Lambda_2 \in (\mathcal{B}, \mathcal{L})$  with  $\Lambda_1 \neq \Lambda_2$  is symbolized as  $\Lambda_1 \rightleftharpoons \Lambda_2$ .
- $\mathcal{K}_{sub} = (\mathcal{B}_{sub}, \mathcal{L}_{sub})$  is a **sub chain** of the chain  $\mathcal{K}$ , if  $(\mathcal{K}_{sub} \text{ is a chain} \wedge \forall \Lambda_{sub} \in \mathcal{K}_{sub} : \Lambda_{sub} \in \mathcal{K}) \Leftrightarrow: \mathcal{K}_{sub} \subseteq \mathcal{K}$ .
- Two chains  $\mathcal{K}_1$  and  $\mathcal{K}_2$  are **equal**, when  $(\mathcal{K}_1 \subseteq \mathcal{K}_2 \wedge \mathcal{K}_2 \subseteq \mathcal{K}_1) \Leftrightarrow: \mathcal{K}_1 = \mathcal{K}_2$ .
- $\mathcal{K}_{rsub} = (\mathcal{B}_{rsub}, \mathcal{L}_{rsub})$  is a **true sub chain** of the chain  $\mathcal{K}$ , when  $(\mathcal{K}_{rsub} \subseteq \mathcal{K} \wedge \mathcal{K}_{rsub} \neq \mathcal{K}) \Leftrightarrow: \mathcal{K}_{rsub} \subset \mathcal{K}$ .
- The **junction** between two chains  $(\mathcal{B}_1, \mathcal{L}_1)$ ,  $(\mathcal{B}_2, \mathcal{L}_2)$  is defined by  $(\mathcal{B}, \mathcal{L}) = (\mathcal{B}_1, \mathcal{L}_1) \cup (\mathcal{B}_2, \mathcal{L}_2) = (\mathcal{B}_1 \cup \mathcal{B}_2, \mathcal{L}_1 \cup \mathcal{L}_2)$ . By definition  $(\mathcal{B}, \mathcal{L})$  is a chain again.

This basic arithmetic allows the chains to be divided into different classes.

### Definition 3 Chain Classification

- A chain  $\mathcal{K} = (\mathcal{B}, \mathcal{L})$  is defined as **true**, if every base is attached to exactly one  $\Omega \in \mathcal{L}$ .
- $\mathcal{K} = (\mathcal{B}, \mathcal{L})$  is defined as **connected**, when  $\mathcal{K}$  is a true chain and  $\forall \Omega_i, \Omega_j \in \mathcal{L} : i \neq j \Rightarrow \Omega_i \rightleftharpoons \Omega_{k_1} \rightleftharpoons \Omega_{k_2} \rightleftharpoons \dots \rightleftharpoons \Omega_{k_m} \rightleftharpoons \Omega_j$  with  $\Omega_{k_p} \in \mathcal{L}$  for  $p = 1(1)m$ .

Now let  $\mathcal{K} = (\mathcal{B}, \mathcal{L})$  be a connected chain:

- $\mathcal{K}$  is defined as **simple chain**, when  $\forall \Omega_i \in \mathcal{L} : \text{there are no more than two } \Lambda_1, \Lambda_2 \in (\mathcal{B}, \mathcal{L}) \text{ with } \Lambda_1 \rightleftharpoons \Omega_i \rightleftharpoons \Lambda_2 \wedge \Lambda_1 \neq \Lambda_2$ .
- $\mathcal{K}$  is named a **closed chain**, when  $\forall \Omega_i \in \mathcal{L} : \exists \Lambda_j, \Lambda_k \in \mathcal{K} \text{ with } \Lambda_j \rightleftharpoons \Omega_i \rightleftharpoons \Lambda_k \wedge \Lambda_j \neq \Lambda_k$ .

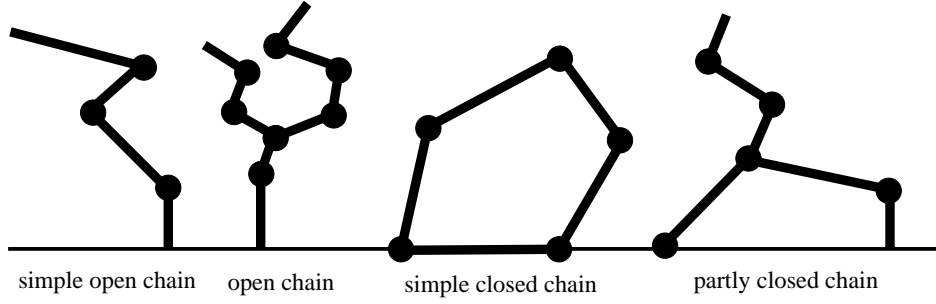


Figure 1: Types of Chains

- $\mathcal{K}$  is defined as **open chain**, when  
 $\nexists \mathcal{U} \subseteq \mathcal{K} : \mathcal{U}$  is a closed chain.
- $\mathcal{K}$  is **partly closed chain**, when  
 $\exists \mathcal{U}, \mathcal{V} \subseteq \mathcal{K} : \mathcal{U}$  is open  $\wedge \mathcal{V}$  is closed  $\wedge \mathcal{U} \not\subseteq \mathcal{V}$ .

#### Theorem 4

Let  $\mathcal{K}=(\mathcal{B}, \mathcal{L})$  be a chain. Then for every  $\Omega \in \mathcal{L}$  exists a simple open chain  $\mathcal{K}' \subseteq \mathcal{K}$  with a minimum number of links connecting the link  $\Omega$  with a base  $B \in \mathcal{B}$ .

#### Proof:

Let  $\mathcal{K}' := (\mathcal{B}', \mathcal{L}') = (\{B\}, \{\Omega_1, \Omega_2, \dots, \Omega_n, \Omega_f\}) \subseteq \mathcal{K}$  be the chain with the minimum number of chains connecting  $\Omega_f$  to the base B. Such a chain exists by Definition 1.

$\mathcal{S} := \{\Omega \mid \Omega \in \mathcal{L}' \text{ with no more than two } \Lambda_k, \Lambda_l \in \mathcal{K}' \text{ attached to } \Omega\}$

Suppose  $\mathcal{K}'$  is not a simple open chain.

$\Rightarrow \exists \Omega_0 \in \mathcal{L}'/\mathcal{S} :$

$\mathcal{K}'' := (\mathcal{B}', \mathcal{L}'') = (\{B\}, \{\Omega_{s_1}, \Omega_{s_2}, \dots, \Omega_{s_m}, \Omega_0\}) \subseteq \mathcal{K}'$

with  $B \rightleftharpoons \Omega_{s_1} \rightleftharpoons \Omega_{s_2} \rightleftharpoons \dots \rightleftharpoons \Omega_{s_m} \rightleftharpoons \Omega_0$  and  $\Omega_{s_i} \in \mathcal{S}$  for  $i = 1(1)m$ .  $\Rightarrow \forall \Omega \in \mathcal{L}''$ ; there are no more than two  $\Lambda_1, \Lambda_2 \in \mathcal{K}'' : \Lambda_1 \rightleftharpoons \Omega \rightleftharpoons \Lambda_2 \wedge \Lambda_1 \neq \Lambda_2$

$\Rightarrow \mathcal{K}''$  is a simple open chain  $\Rightarrow \mathcal{K}'' \subset \mathcal{K}' \Rightarrow \Omega_f \notin \mathcal{K}''$  because otherwise  $\mathcal{K}'$  is not the chain with a minimum amount of links.

Define  $\mathcal{C}_{\Omega_0} := \{\Omega \in \mathcal{L}'/\mathcal{L}'' \mid \Omega \rightleftharpoons \Omega_0\} = \{\Omega_{c_1}, \Omega_{c_2}, \dots, \Omega_{c_p}\}$   
 $\Rightarrow p \geq 2 \Rightarrow \forall \Omega_{c_i} \in \mathcal{C}_{\Omega_0} : (\mathcal{B}', \mathcal{L}'/\{\Omega_{c_i}\}) \not\subseteq \mathcal{K}'$  and  $(\mathcal{B}', \mathcal{L}'' \cup \{\Omega_f\}) \not\subseteq \mathcal{K}'$  because otherwise  $\mathcal{K}'$  is not the chain with a minimum amount of links  $\Rightarrow \Omega_f \notin \mathcal{C}_{\Omega_0}$ .

Now let  $\mathcal{M}_{\Omega_{c_i}} :=$

$\{\Omega \in \mathcal{L}'/\{\mathcal{L}'' \cup \mathcal{C}_{\Omega_0}\} \mid \exists \Omega_{m_1}, \Omega_{m_2}, \dots, \Omega_{m_q} \in \mathcal{L}'/\{\mathcal{L}'' \cup \mathcal{C}_{\Omega_0}\} :$

$\Omega \rightleftharpoons \Omega_{m_1} \rightleftharpoons \Omega_{m_2} \rightleftharpoons \dots \rightleftharpoons \Omega_{m_q} \rightleftharpoons \Omega_{c_i}\}$ .

$\Rightarrow \forall \Omega_{c_i} \in \mathcal{C}_{\Omega_0} : \Omega_f \notin \mathcal{M}_{\Omega_{c_i}}$ , otherwise

$\Omega_f \in (\mathcal{B}', \mathcal{L}'' \cup \{\Omega_{c_i}\} \cup \mathcal{M}_{\Omega_{c_i}}) \subset \mathcal{K}'$

$\Rightarrow \Omega_f \notin \bigcup_{\Omega_{c_i} \in \mathcal{C}_{\Omega_0}} \mathcal{M}_{\Omega_{c_i}} = \mathcal{L}'/\{\mathcal{L}'' \cup \mathcal{C}_{\Omega_0}\}$

$\Rightarrow \Omega_f \notin (\mathcal{B}', \mathcal{L}'' \cup \mathcal{C}_{\Omega_0} \cup \mathcal{L}'/\{\mathcal{L}'' \cup \mathcal{C}_{\Omega_0}\}) = \mathcal{K}'$

(Contrary to  $\Omega_f \in \mathcal{K}$ )

$\Rightarrow \mathcal{K}'$  must be a single open chain. qed.

**Note:** Theorem 4 implies that every true chain can be composed by a finite number of single open chains. Therefore it is enough to concentrate on open chains.

The sets of links and bases will be more closely examined in the next section.

### 3.1 Links

To represent a robot by an open chain, attributes have to be defined for every link  $\Omega$ . This section investigates the attributes needed to “bring life” to this structure.

- As mentioned in Definition 1, every link is attached through a joint with another link or base. Therefore a link and its joint should be seen as one unit. This is represented by the joint value  $j_{value}$  which describes the movement along or parallel to a plane.
- Open chains can also have multiple successors (see Fig 1). Thus the structure  $\Omega$  should also include a set  $\mathcal{S}$ , listing all the successors and certain attributes defining the relationship between the link and its successors, e.g. DH-Parameters [2].
- The successors should be listed by a property resembling the link in the chain. Making a link unique to any other, a name  $l_{name}$  is attached to the structure  $\Omega$ .
- The link characteristics also include a set  $\mathcal{D}$  defining whether the joint movement depends on other joint movements or not.
- For further extensions, like minimum and maximum joint values, a set  $\mathcal{E}$  must be added to the structure.

### Definition 5 The Link Structure

The link structure  $\Omega_i$  of the chain

$\mathcal{K} = (\mathcal{B}, \{\Omega_1, \Omega_2, \dots, \Omega_i, \Omega_{i+1}, \dots, \Omega_n\})$  is defined by  $\Omega_i := (l_{name}, j_{value}, \mathcal{D}, \mathcal{E}, \mathcal{S})$ , with

- $\mathcal{D} := \{\mathcal{D}_{name}, d_{fct}(x)\}$  where  $\mathcal{D}_{name} = \{d_1, d_2, \dots, d_m\}$  listing the links  $j_{value}$  depends on, and  $d_{fct} : \mathbb{R}^m \mapsto \mathbb{R}$ ,  $x \mapsto d_{fct}(x)$  describing the relationship between  $j_{value}$  and the other joint values.  
If  $\mathcal{D} = \{\}$  then the link is independent.
- $\mathcal{S} := \{(n_1, a_1), (n_2, a_2), \dots, (n_i, a_i)\}$ , where  $n_i$  is the successor's name and  $a_i$  the attributes defining the relation between  $\Omega_i$  and  $n_i$ .

### Definition 6 End Effector

Successors of link  $\Omega$  or a base  $B$ , which do not resemble a link, are called **end effectors**. They are neither a base nor a link. Therefore, they are not elements of the chain. An end effector resembles a location defined by its purpose and the relation to its predecessor. Depending on this purpose, end effectors receive predefined names :

- *RTCP (Real Tool Center Point)*  
another chain can be mounted at this position
- *VTCP (Virtual Tool Center Point)*  
no mounting is possible at this end effector.

Therefore, end effectors clearly define locations in a chain where other chains can be mounted on. The example in section 3.3 will clarify the attachment of a chain to another.

### 3.2 Bases

The base  $B_i \in \mathcal{B}$  of a chain  $\mathcal{K} = (\mathcal{B}, \mathcal{L})$  defines the relationship between a fixed global system and a link or an end effector, i.e. the position of a link to a global coordinate system. Therefore, the base structure should be defined by  $b_{name}$ , the corresponding link or end effector name, and  $b_{att}$ , the attributes describing the relation between  $b_{name}$  and the global system.

### Definition 7 The Base Structure

The base structure  $B_i$  of the chain

$\mathcal{K} = (\{B_1, B_2, \dots, B_i, B_{i+1}, \dots, B_t\}, \mathcal{L})$  is defined by  $B_i := (b_{name}, b_{att})$ .

### 3.3 Example

The manipulator M1 consists of three bases, two links “left” and “right” and an end effector “VTCP”

(see Figure 2). These links move synchronously with the in- and outside. Therefore, the joint value of link “left” depends on the joint value of link “right”. M1 is being represented in the previously developed structure by

$$\mathcal{K}_1 = ( \{ ( \text{right}, b_{att_{right}}, ( \text{VTCP}, b_{att_{VTCP}}, ( \text{left}, b_{att_{left}} ) \} ), \{ ( \text{right}, 28.5, \{ \}, \mathcal{E}_{right}, \{ \} ), ( \text{left}, -28.5, \{ \text{right}, -x \}, \mathcal{E}_{left}, \{ \} ) \} ) ).$$

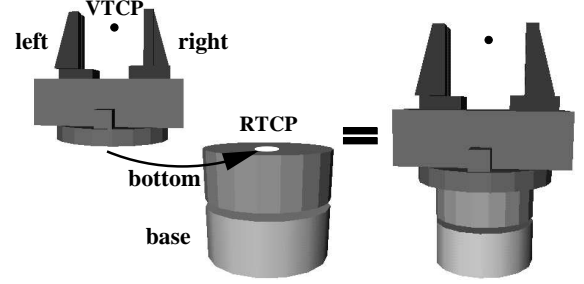


Figure 2: Manipulator M1 mounted on M2

The second manipulator M2 is defined by a base, a link “bottom” and an end effector “RTCP”. Thus the chain representing M2 looks like

$$\mathcal{K}_2 = ( \{ ( \text{bottom}, b_{att_{bottom}} ), \{ ( \text{bottom}, 0, \{ \}, \mathcal{E}_{bottom}, \{ ( \text{RTCP}, a_{RTCP} ) \} ) \} ).$$

For a better understanding of the structure’s flexibility manipulator M1 (represented by  $\mathcal{K}_1$ ) is mounted on the RTCP of M2 (represented by  $\mathcal{K}_2$ ). Thereby a function determines the attribute’s values defining the relations between “bottom” and “right”, “VTCP” and “left”. Thus a new chain  $\mathcal{K}$  is created:

$$\mathcal{K} = ( \{ ( \text{bottom}, b_{att_{bottom}} ), \{ ( \text{bottom}, 0, \{ \}, \mathcal{E}_{bottom}, \{ ( \text{right}, a_{right} ), ( \text{left}, a_{left} ), ( \text{VTCP}, a_{VTCP} ) \} ), ( \text{right}, 28.5, \{ \}, \mathcal{E}_{right}, \{ \} ), ( \text{left}, -28.5, \{ \text{right}, -x \}, \mathcal{E}_{left}, \{ \} ) \} ).$$

It is possible to separate  $\mathcal{K}$  as easily as it is to assemble it.

This structure enables simulations to work with very complicated structures. An example is the simulation  $KL_iMt$  (Kinematic Library for Simulating Manipulators): a general approach to simulate the behaviour of various types of manipulators.

## 4 The Robot Library KLIMT

Based on the model introduced in the previous section 3, the user interface was developed (see Figure 3).

Following the formal specification, the interface helps assemble chains very easily. However, the user does not need to know about the chain structure explained in section 3. Graphical dialogues prompt the user for the specific data that is required for the final model. This includes the specification of links, bases, VTCPs and RTCPs.

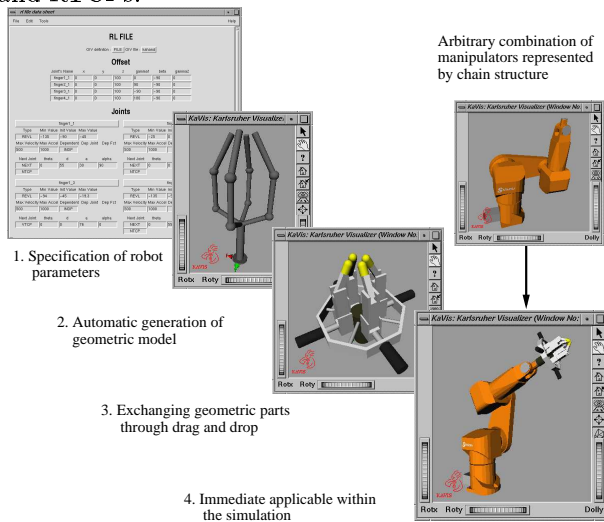


Figure 3: *Creating a model with Klim*

At this point, the library provides several algorithms for the inverse kinematics, interpolation for trajectories (point to point and straight line) and the calculation of velocity profiles. The inverse kinematic algorithm is based on an iterative approach [9]. New algorithms have been developed to improve the performance of this method [7].

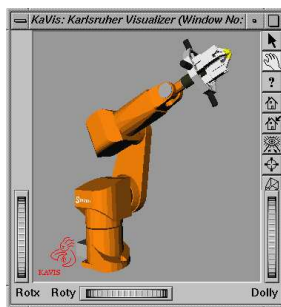


Figure 4: *Traditional 6-axis robot with complex 4 finger gripper.*

The user is faced with two possibilities of how to include a model to the library.  $KL_iMt$  is capable of reading CAD-files based on VRML or INVENTOR. It can transform the geometric data into the described chain structure. However, certain joint specifications,

e.g. dynamic parameters, cannot be extracted out of a geometric model. Thus the user himself/herself has to specify these through the user interface. The robot can then be controlled and visualized in the simulation environment KAVIS [8].

Another possibility is described in Figure 3. In this case the user only knows the kinematic and dynamic parameters. Through the user interface the simulation will ask for all specifications needed. Afterwards  $KL_iMt$  can generate a geometric model. In the third step parts of the model can be replaced with more realistic counterparts. Due to the general approach of the simulation, the full functionality of  $KL_iMt$  can be used for the model, e.g. mounting on other manipulators or moving it around.

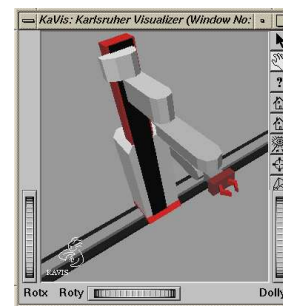


Figure 5: *Robot with rotatory and translatory joint with 7 degrees of freedom*

Figures 4, 5 and 6 show examples that are included in the simulation package. Although the robot's designs are completely different, w. r. t. joint characteristics and physical structure (gripper, manipulator, mobile system) their kinematic and dynamic behaviour can be represented in the displayed *chain structure*.

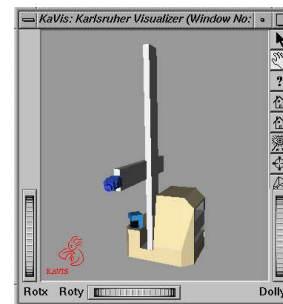


Figure 6: *Mobile robot with one arm manipulator (6 degrees of freedom)*

In addition to a user interface to control models, the library provides a protocol interface based on PVM

(Parallel Virtual Machine)[3] to allow access for several client processes. Thus, the system can very easily be included in other applications.

To summarize, the robot library  $KL_iMt$  provides the following features:

- Easy modeling of robotic features through formal structure.
- Arbitrary connection between chains (like the example in section 3.3).
- Simple integration of VRML or INVENTOR based CAD-models. Thus, geometric based simulation results may be obtained, e.g. collision detection, object handling, etc..
- Powerful algorithms for inverse kinematics and interpolation of trajectories.
- Test and validation of several algorithms for kinematic and dynamic calculations.
- Interface for client applications that use the library's functionality.

## 5 Conclusion

In this paper a formal definition for robot manipulators has been presented. The concepts have been implemented in the robot library  $KL_iMt$  which allows easy construction and simulation of arbitrary types of manipulators and grippers.  $KL_iMt$ 's interface also allows remote client applications to use the library's functionality. Additionally, robot and gripper types can be adjoined very simply. The iterative algorithms which solve the inverse kinematic and dynamic problems make  $KL_iMt$  a powerful software tool that allows easy testing of prototype robots for arbitrary tasks.

## Acknowledgements

This work has partially been supported by the *Deutsche Forschungsgemeinschaft* project "Programmieren durch Vormachen" and the BRITE-EuRAM project 1564 "Skill-MART". It has been performed at the Institute for Real-Time Computer Systems & Robotics, Prof. Dr.-Ing. H. Wörn, Prof. Dr.-Ing. U. Rembold and Prof. Dr.-Ing. R. Dillmann, Department of Computer Science, University of Karlsruhe.

## References

- [1] Peter Corke. A robotic toolbox for matlab. *IEEE Robotics and Automation Magazine*, 3(1):24–32, March 1996.
- [2] J. Denavit and R.S. Hartenberg. A kinematic notation for lower pair mechanisms based on matrices. *Journal of Applied Mechanics*, 22:215–221, 1955.
- [3] Al Gheist, Adam Beguelin, Jack Dongarra, Weicheng Joang, and Robert Manchek an Vaidy Sunderam. *PVM: Parallel Virtual Machine : A Users' Guide and Tutorial for Networked Parallel Computing*, 1994.
- [4] Richard Gourdeau. Object oriented programming for robotic manipulators simulation. *IEEE Robotics and Automation Magazine*, 4(3):21–29, September 1997.
- [5] S. Owen, M. C. Bonney, and A. Denford. A modular re-configurable approach to the creation of flexible manufacturing cells for educational purposes. In *Proceedings of the International Conference on Intelligent Robots and Systems*, volume 3, pages 1369–1374, Grenoble, France, September 1997.
- [6] Aurelio Piazzini and Antonio Visioli. A global optimization approach to trajectory planning for industrial robots. In *Proceedings of the International Conference on Intelligent Robots and Systems*, volume 3, pages 1553–1559, Grenoble, France, September 1997.
- [7] Kilian M. Pohl. Modellierung von Manipulationssystemen. Master's thesis, University of Karlsruhe, 1998.
- [8] Horst F. Schaudte. *Überprüfung und Überwachung in der Telepräsenz*. PhD thesis, University of Karlsruhe, Dezember 1997.
- [9] L.-C. T. Wang and C.C. Chen. A combined optimization method for solving the inverse kinematics problem of mechanical manipulators. *The International Journal of Robotics Research*, 1991.