

Institute of Parallel and Distributed Systems  
University of Stuttgart  
Universitätsstraße 38  
D-70569 Stuttgart

Bachelor Thesis Nr. 2353

# **Analysis of Kinematics and Dynamics of Modular Robot Assemblies**

Jutta Ganzhorn

**Course of Study:** Engineering Cybernetics  
**Examiner:** Prof. Dr. rer. nat. habil. Paul Levi  
**Supervisor:** Dipl.-Ing. Eugen Meister

**Commenced:** 15.07.2011

**Completed:** 15.11.2011

**CR-Classification:** I.2.2, I.2.6, I.2.9, I.2.11

# Acknowledgements

I would like to express my gratitude to everyone who helped me in the commencement and completion of this work.

This thesis would not have been possible without Prof. Dr. rer. nat. habil. Paul Levi granting me the opportunity to broaden my knowledge in modular robotics and submit my Bachelor Thesis in his institute.

I wish to acknowledge the support from Dipl.-Ing. Eugen Meister in his role as supervisor. I appreciate his openness to my ideas and his suggestions.

I would like to express my gratitude to everybody who supported me during the past months.

Last but not least, I take this opportunity to express my gratitude to my parents, without whose support my studies and stays abroad would not have been possible.

## **Abstract**

The development of self-reconfigurable modular robots has experienced significant progress. Continuous improvement during the past twenty years produced flexible and easy maintainable robot mechanisms. Self reconfiguring modular robots consist of various similar robotic modules and are designed to establish manifold connections between each of these link modules. Moreover these modules are able to perform movements to change the shape or the position of the robotic chain.

In this thesis the main focus lies on the analysis of kinematics and dynamics of small modular robot organisms and their movement pattern. Although diverse linkage mechanisms do exist, practically no motion of one module can be accomplished without interacting with a second one. Therefore furthermore fundamental is the study and examination of dyad kinematics which represent the pattern of movement between solely two modules. However generation and simulation of models of modular robot kinematics and dynamics are complex and manual derivation needs tremendous efforts as every configuration and alternation of shape induce new changes of parameters' values. Besides machine-driven computation requires a great deal of energy and increased storage capacity if every module constellation and applicable movements are predefined in a database. To avoid this squandering of resources a framework is implemented in Matlab based on Chen's theory to design an accurate dynamic model.

In the end, this thesis shall provide a farther step towards a robust and flexible modular robot organism which is able to perform reliable interaction with the environment to constitute to an optimized and effective realization of tasks assigned to it in industry or in the future private households.

# Contents

<b>List of Figures</b>	<b>V</b>
<b>List of Tables</b>	<b>VII</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	2
1.2 Goals . . . . .	2
1.3 Current and Related Projects . . . . .	4
1.4 Organization of this thesis . . . . .	5
<b>2 Foundations</b>	<b>6</b>
2.1 General Description of Module Design in Robotics . . . . .	6
2.1.1 Module Assemblies . . . . .	8
2.1.2 Module Traversing Order . . . . .	9
2.2 Screw Theory . . . . .	10
2.3 Geometric Formulation of Modular Robots . . . . .	13
2.3.1 Connectivity Matrix . . . . .	13
2.3.2 Accessibility Matrix . . . . .	14
2.3.3 Assembly Incidence Matrix . . . . .	16
2.3.4 Path Matrix . . . . .	19
<b>3 Representation of Multi-Robot Organisms</b>	<b>20</b>
3.1 Adapted Module Design . . . . .	20
3.2 Adapted Assembly Incidence Matrix . . . . .	23
3.2.1 Joint Axes Configurations . . . . .	26
<b>4 Implementation</b>	<b>32</b>
4.1 Automatic Model Generation . . . . .	32
4.1.1 Dyad Kinematics . . . . .	32
4.1.2 Forward Kinematics . . . . .	35
4.1.3 Dynamics . . . . .	36
4.1.4 Comparison of Modular Robot Assembly Prototypes . . . . .	44
<b>5 Conclusion</b>	<b>50</b>
5.1 Summary . . . . .	50
5.2 Further research . . . . .	51
<b>Bibliography</b>	<b>A</b>



# List of Figures

1.1	Humanoids . . . . .	1
1.2	Automatic Model Generation . . . . .	3
1.3	Modular Snake Robots . . . . .	4
2.1	Two different Module Types both in Dyadconfiguration:(a)[iMobot, 2011],(b) [CKBot, 2009] . . . . .	7
2.2	Revolute Joint and its Rotation Possibility . . . . .	7
2.3	Prismatic Joint and its Translation Possibility . . . . .	8
2.4	Virtual Joint neither Translation nor Roatation Possibilities . . . . .	8
2.5	Modular Robot Assemblies: Serial Type . . . . .	9
2.6	Modular Robot Assemblies: Branched Types . . . . .	9
2.7	Twist: Pair of angular and linear velocity vector. . . . .	12
2.8	Wrench: Pair of force and torque vector. . . . .	12
2.9	Modular Robot Assembly with $n = 9$ Modules and $n - 1 = 8$ Joints. . . . .	14
2.10	The Directed Kinematic Graph $\vec{\mathcal{G}}$ with 9 Vertices $v_i$ and 8 Edges $e_j$ of the given robot. . . . .	15
2.11	Example of the Modular Robot with $n = 9$ Modules and $n - 1 = 8$ Joints and the belonging Accessibility Matrix $\mathcal{R}(\vec{\mathcal{G}})$ . . . . .	16
3.1	Scout and Backbone cube modules developed in SYMBRION and REPLICATOR . . . . .	20
3.2	Backbone Platform: Front and Right Side. [Kernbach et al., 2011] . . . . .	21
3.3	Scout Platform . . . . .	22
3.4	Integrated Rotary Joints . . . . .	23
3.5	Modules Labeled by L2 . . . . .	24
3.6	scoutdyadlabeled . . . . .	24
3.7	Dyad Configurations $\mathcal{SO}$ . . . . .	29
3.8	Dyad Configurations $\mathcal{SP}$ . . . . .	30
3.9	Dyad Configurations $\mathcal{SS}$ (1) . . . . .	30
3.10	Dyad Configurations $\mathcal{SS}$ (2) . . . . .	31
3.11	A serial typed robot consisting of $n = 3$ modules and its $\mathcal{AAM}$ . . . . .	31
4.1	Dyad Kinematics . . . . .	33
4.2	TreeRobotKinematics Algorithm. . . . .	36
4.3	DynamicModelGeneration . . . . .	43

# List of Tables

3.1	Look-Up Table for Dyads labeled by $\mathcal{L} \in$ and their Axes Constellations. . .	28
4.1	Serial Type Modular Robot Assembly with $n = 3$ Modules. . . . .	44
4.2	Branching Type Modular Robot Assembly with $n = 4$ Modules. . . . .	45
4.3	Branching Type Modular Robot Assembly with $n = 5$ Modules. . . . .	46
4.4	Branching Type Modular Robot Assembly with $n = 5$ Modules. . . . .	47
4.5	Dynamics and Kinematics of a Dyad with $n = 2$ Modules. . . . .	49

# 1 Introduction

The world of Robotics has to live up to the ideals of humanoid robots due to exaggerated expectations of ever improving and impressively looking robots communicated by media and yet achievements can be seen. However not only regarding the appearances accomplishments are in process.

In the last two decades a tremendous effort has been made to an augmentation of functionality. Since it is a well known fact regarding society that there is strength in numbers, cooperation between robots gains importance in science every day. Hoping to implement team work and collaboration for a more effective working routine in industry, in the last twenty years researchers' focus was drawn towards reconfigurable modular robots.

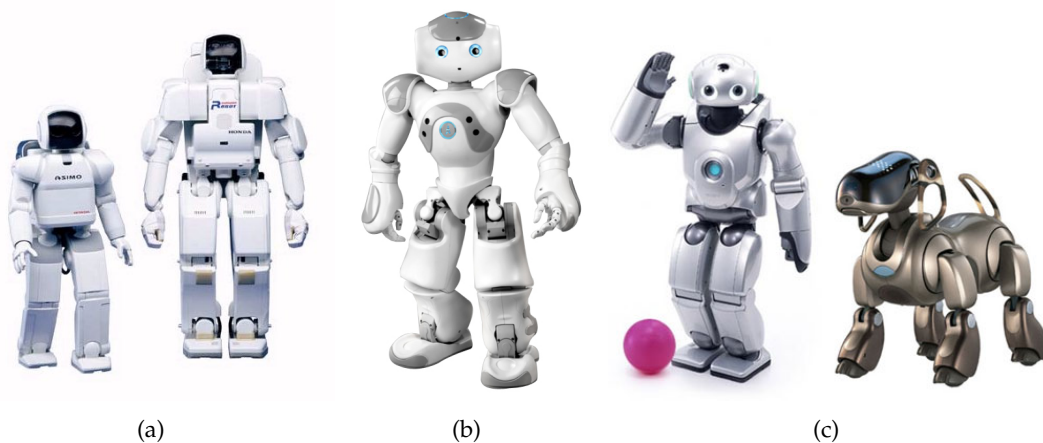


Figure 1.1: Humanoid Robots: (a) Asimo [Asimo, 2004]. (b) Aldebaran Nao H21 [Aldebaran, 2010]. (c) Qrio with Dog Aibo [Qrio, 2006]

Robots automation and electro-mechanical devices came a long way until they achieved indispensability in several industry sectors. But also concerning several environments where danger or health damaging circumstances prevent mankind from proper independent reliable work performance and hiring of handcrafts is irresponsible, robot manipulators are involved. Even in medical issues where humans' preciseness is not serving the need, perfectly trimmed computer-controlled mini robots can assist.

In these named areas collaboration enlarges the spectrum of robot application. This is why currently the attention of researchers and the development at major institutes increasingly comprise multiple cooperating robots up to the innovation of modular robots, which bring along huge advantages for a promising future.

## 1.1 Motivation

The elegance and benefits of modular robot assemblies lie in their reconfiguration and simplicity which guarantee that all modules are replaceable. If complications occur, problems are easily specified and solved as only the affected module has to be compensated by a second similar module, thus maintenance is easy. But the principal effect of modular robot is their adaptability to optimize their functionality by altering their shapes for according to the tasks.

Modular robots are assemblies of smaller or bigger chains of modules. Modules can be links or joints to establish a wide range of different robot geometries. This allows the system to cope flexibly with any solicited task and supplies the assembly configuration with agility.

Intuitively motion needs strict control mechanisms in order to obtain desired movement pattern. Obviously not only control experiments can lead to significant results but also computational simulation plays an important role.

To receive trustworthy simulation results the precise modeling is the basis component. Dynamical and Kinematical models of the robot assemblies have to be detected.

Nonetheless classical dynamical model generation does not include ever changing modular assemblies. It is rather established for robots with fixed geometrical appearances, where models can be derived a priori of the simulation.

These deriving techniques cost much time and effort as they have to be done manually or can be provided by commercial software packages with predefined models of existing robots. This explains why the common approach of model generation is not appropriate for the modular robot assemblies. Modular robots do not suit any predefined patterns as the possible robot geometries can adapt endless different shapes. Software packages would need huge libraries of robot configurations and computer would lack of disk storage capacity and efficiency would not be granted, if conventional modeling techniques were used.

This thesis deals with the automatic model procedure for modular robot assemblies for control and simulation purposes based on screw theory.

## 1.2 Goals

As can be seen in the former section model generation for modular reconfigurable robot dynamics and kinematics can be hard work. To avoid complicated techniques an easy framework for modular robot is examined to obtain the needed models for the dynamics and kinematics computation.

An anterior endeavor to handle automatic model generation for modular robots was the Denavit-Hartenberg Parameterization.

Denavit-Hartenberg convention is an approach introduced by Richard S. Hartenberg

and Jacques Denavit to find good coordinate frames of robotic configurations. Denavit-Hartenberg describes the motion of a series of rigid links connected by various joints to obtain forward and inverse kinematics. Two adjacent modules are considered. The first one is called the previous whereas the second is the actual module.

Selecting frames of references is based on basic transformations to allow a minimal representation. There are four parameters, which need to be defined, to specify the transformation between two coordinate systems:

- $d$ : is the depth along the previous actuator axis from its origin to the normal. Link offset.
- $r$ : is the distance along the rotated x-axis. Alternatively it is spoken as radius of rotation about previous z-axis. Link length.
- $\alpha$ : rotates about the new x-axis to bring z into line with its desired position. Link twist.
- $\theta$ : rotates along the previous z-axis to align the x-axis. Thus it is the angle about the previous z to align the previous x with the new origin. Joint angle.

However the automatic model generation including Denavit-Hartenberg parameters does not distinguish precisely between the spatial relation and the arranging module's sequence in the robot incidence. Moreover a different initial or zero position causes dissimilar assortment of parameters, as the Denavit-Hartenberg method depends on it. Also the smallest of the deviance of parallelism bring about a new set of parameters.

Therefore the following framework shall be independent of the robot's actual geometry to save storage space and be more effective.

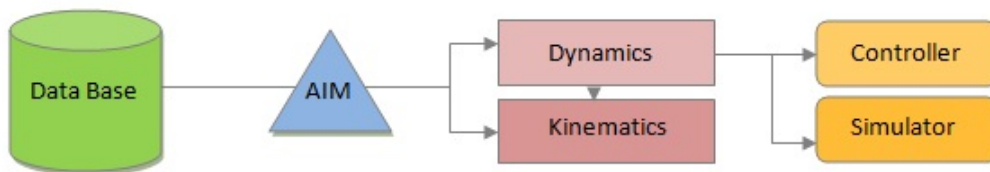


Figure 1.2: Automatic Model Generation [Chen and Yang, 1998].

Figure 1.2 presents the general design of the framework of the general model approach is shown in [Chen and Yang, 1998]. The authors describe the main components as the following:

- Data Base  
The Data Base is referring to a library where all the possible components of a modular robot assembly are stored. Furthermore it contains all the specifications and description of the basic components. For example actuators, sensors end-effectors can be found in this component database.

- Assembly Incidence Matrix

The Assembly Incidence Matrix (AIM) is very important for this modeling technique as it describes the current modular robot assembly based on a kinematic graph. The AIM shows the different linkage and connection mechanisms and conveys type and sequence of the actual robot incidence, without changing other parameters modules can be attached or detached.

- Modeling Techniques for Kinematics and Dynamics

It is very important to stress that the Modeling Techniques for Kinematics and Dynamics are geometry. Therefore the kinematics and dynamics are represented by Lie Groups and Lie Algebra to provide the independence of the joint types.

As Chen et al. developed the theoretical framework, this thesis is aimed at actually implementing it and providing a further step towards modular robot with full functionality.

### 1.3 Current and Related Projects

Various research institutes developed modular robot systems and manufactured prototypes. Figure 1.3 supplies examples.

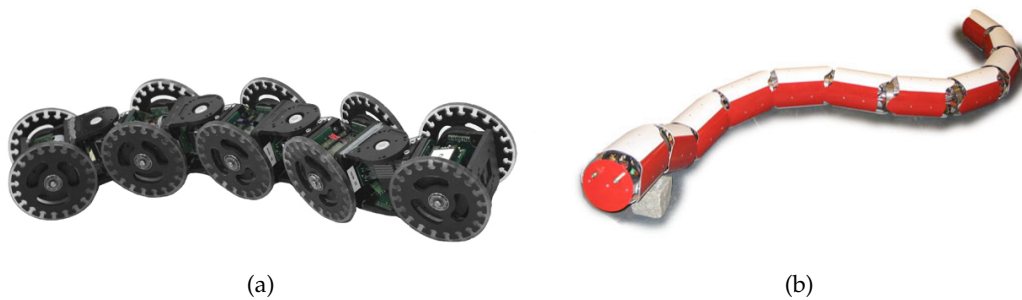


Figure 1.3: Modular Snake Robots: (a) CKBot [CKBot, 2009]. (b) Anna Konda [AnnaKonda, 2006].

The Modular Robotics Lab (ModLab), a subgroup of the GRASP Lab and the Mechanical Engineering and Applied Mechanics Department at the University of Pennsylvania developed the CKBot (Connector Kinetic roBot) module to be fast, small, and inexpensive. [CKBot, 2009]

Anna Konda is a water-powered fire fighting snake robot developed by The Foundation for Scientific and Industrial Research at the Norwegian Institute of Technology (SINTEF). Further Reading on [AnnaKonda, 2006]

This thesis is not based on related projects but contains slight similarities to the two projects SYMBRION (Symbiotic Evolutionary Robot Organisms) and REPLICATOR (Robotic Evolutionary Self-Programming and Self-Assembling Organisms) which are sponsored

by the European Commission.

The main focus of these projects is to investigate and develop novel principles of adaptation and evolution for symbiotic multi-robot organisms based on bio-inspired approaches and modern computing paradigms.

The REPLICATOR focuses inter alia on intelligent, reconfigurable and adaptable "carrier" of sensors (sensors network), extremely powerful computational on-board resources and systems with a large number of light modules or medium number of heavy modules. Therefore the thesis can be rather correlated with the latter project. Further information can be found on [SYMBRION, 2008].

### 1.4 Organization of this thesis

This thesis consists of 5 Chapters:

- Chapter 1 represents the general introduction into the multi robot organisms and the correlated problems. It comprises motivation and goals of this thesis and gives hints about former, present and future work or projects.
- Chapter 2 is about the theoretical foundations of the automatic generation of kinematics and dynamics models: module design, screw theory and the basics of geometric formulations of modular robot assemblies.
- Chapter 3 explains the differences between theory and adapted practically applied methods.
- Chapter 4 contains the implementations developed during this thesis. All implementations are employed in MatLab.
- Chapter 5 provides the conclusion, where the most important items are stressed and furthermore gives hints for further studies.

## 2 Foundations

In this chapter the theoretical foundations of module design in robotics are described. Simultaneously it pictures module assemblies in general as well as module assemblies in general. Furthermore the principle pillars of the screw theory, the mathematical background of the dynamics and kinematics description, are regarded. In detail the principle pillars of geometric representation are explained with special focus on the relevant matrices for the automated model generation for modular robot dynamics.

### 2.1 General Description of Module Design in Robotics

The trend in the last two decades has been towards robots that are applicable to a huge range of diverse assignments of tasks. Beyond that the robots need to be flexible and robust to interact safely in their ever changing environment. Therefore a new concept of robots has been initiated which represents a powerful method to adjust a robot according to the expectations and intentions of the instructor.

This method comprises a new constructing describing method called module set and was introduced by [Chen and Yang, 1997]. They had the purpose to improve production and proposed the reconfigurable modular robot system which is constructed of various modules. These robot systems can vary in various ways. Theoretically there exist no restrictions regarding the number of module or the connecting interaction between those modules. The characteristics of the modules differ slightly from one research center to another. However, they have all these features in common: Modules are non complex, simple, cheap and easy to maintain structures with multiple connecting ports.

Due to the theory of [Chen and Yang, 1998], which is implemented in this bachelor thesis, a module can be one out of three different type sizes but should be cubic to guarantee a maximum of possible connection surfaces. The interaction between the modules can take place in one out of three modes. The rigid, the revoluted and the prismatic connecting ports form the named three types. The connection between two modules always is build up by so called connectors.

Optimally a robot system consists of a minimal number of modules to live up to the expectations brought up with its task that has to be completed. This induces efficiency and energy reduction through a minimized weight. Generally with this condition in mind the bandwidth of the system will enlarge and celerity will increment.

Both connectors and the linkable modules are named modules. To obtain an easy way to differentiate between those two kinds of elements the following nomenclature is recommended. Modules are considered as links if they consist of connecting ports but need





Figure 2.1: Two different Module Types both in Dyad configuration:(a)[iMobot, 2011],(b) [CKBot, 2009]

a connector to accomplish a connection in contrast to connectors which are described by joints. The names are derived of the kinematic features of the module types. The connecting ports of a module are those sides of it, which are able to establish a connection of power, mechanics and control. Therefore the perfect conditions are symmetrical shapes and multiple accesses to synthesize connections. All these requirements are met by the cubic module. As mentioned above cubes are the easiest and miscellaneously deployable elements in module sets. Until today joints are restricted to a maximum of one degree-of-freedom if a single one is considered. The different connectors' features are described in the following:

- **Revolute joints:**

The connectors receive the name revolute joint if they are connected to the surfaces of two modules and cause those sides to rotate against each other. So they are the rotary joints, which rotate along their joint axis.



Figure 2.2: Revolute Joint and its Rotation Possibility

- **Prismatic joints:**

On the other hand prismatic joints exist which produce a linear translation between two connected modules. These joints move linearly along their translation axis.

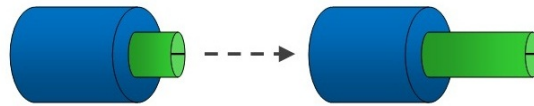


Figure 2.3: Prismatic Joint and its Translation Possibility

- **Virtual joints:**

The use of virtual joint does not allow any joint displacement at all, so it is a rigid connection between two link modules and is used whenever no movement between two modules should take place.

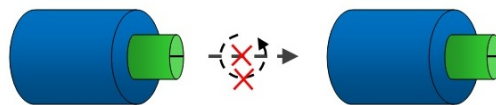


Figure 2.4: Virtual Joint neither Translation nor Roatation Possibilities

### 2.1.1 Module Assemblies

The idea of modular robots brings with it the opportunity of any possible configuration and sequence of modules. This means that an immense spectrum of assembly constellations have to be reduced to reasonable and clear subassembly groups. Therefore two main classes are distinguished within modular robot assemblies, the serial and the tree typed robot assembly. The idea of modular robots brings with it the opportunity of any possible configuration and sequence of modules. This means that an immense spectrum of assembly constellations have to be reduced to reasonable and clear subassembly groups. Therefore two main classes are distinguished within modular robot assemblies, the serial and the tree typed robot assembly. The former refers to an ordinary chain including one starting and one final point connected through a simple consecution of chain links. The latter represents not only one simple chain but a more complicated structure. Starting from one module it can diverge into diverse branches leading to various endpoints. Thus the tree typed robot assembly often is also referred to as the so called

branching type.

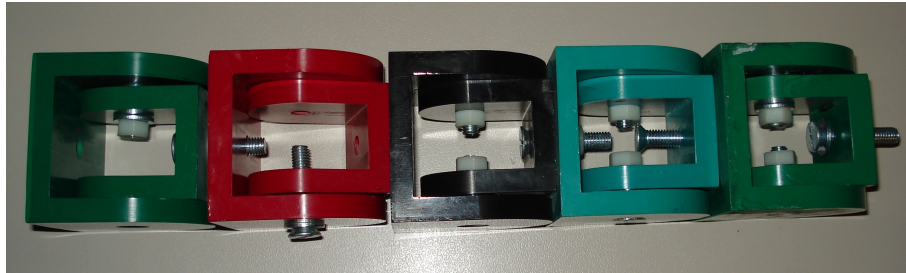


Figure 2.5: Modular Robot Assemblies: Serial Type

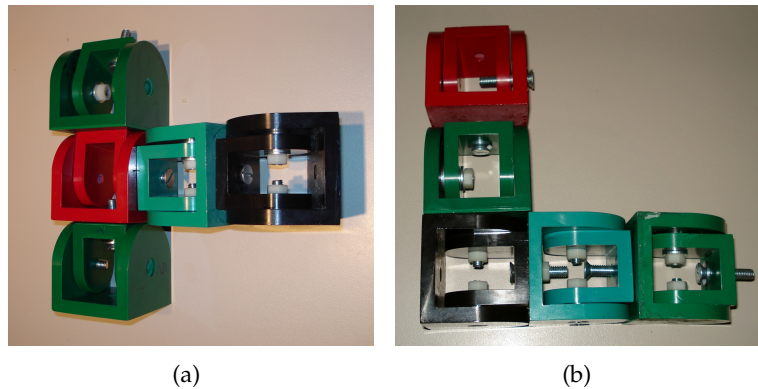


Figure 2.6: Modular Robot Assemblies: Branched Types (b) . (a)

### 2.1.2 Module Traversing Order

To obtain a mechanism that delivers the order of the assembled link modules the module traversing order is of utmost significance. The module traversing order defines the sequence in which the modules receive their label. Labels designate an individual number to each module to differentiate between them effortlessly. This assignment will gain importance in the automatic generation of a model for dynamics of a modular reconfigurable robot.

The two characterized groups, the serial and the branching types, need to be considered in a slightly different way. As the former refers to a normal chain its module traversing order can follow the typical order from its starting point to the end of the chain. Different search algorithms like the Depth-First-Search or Breadth-First-Search deliver the same results. But this does not apply to the later group, the tree typed robot assemblies, which contain diverse branches and tips. This is why the order numbers of the link and joint

modules depend on the graph's searching algorithm. In this case only the Depth-First-Search and the Breadth-First-Search are analyzed to determine the traversing order.

- **Breadth-First-Search**

*Breadth-First-Search* is known as a graph search method. The algorithm defines a root node and always starts at this elected point with exploring all its neighbor nodes. After finishing exploring them, it begins to explore each of their yet unexplored adjacent nodes. This will be repeated until the goal node is found.

- **Depth-First-Search**

*Depth-First-Search* is also a traversing method of graphs. As the Breadth-First-Search the algorithm begins at the node which was selected as the root. Then the exploration continues on the branch of the nearest node as long as it finds the goal node or until it is at the end of the branch. If it does not find a child node, the backtracking will be employed until a unexplored branch is found to be the next to be explored.

So in conclusion it can be said: The farther from the base module the higher is the number of the vertices in the graph considered with Depth-First-Search, whereas Breadth-First-Search is almost the contrary. Breadth-First-Search will compute a totally different traversing order. But both search methods do have in common that their first fixed module always is declared as the base module.

More information about the mentioned fields of module sets and module traversing order can be found in [Deo, 1974], [Chen and Yang, 1997] and [Wurst, 1986] or in the basic works as stated above.

## 2.2 Screw Theory

The idea of screw theory is not new. It was first developed by Sir Robert Stawell Ball in 1876 for application in kinematics and statics of mechanisms [Ball, 1876] but it sank into oblivion. Today it reacquires and increases its importance in Robotics due to excellent application possibilities and results in model generation compared to common methods since screw theory simplifies calculations based on geometries in a three dimensional space.

An adequate method to cope with the problems that occur during the calculations of kinematics and dynamics of rigid bodies is screw theory. It considers the resulting translational and angular velocities supplementary to the emerging forces and moments as pairs of vectors. Generally screw theory displaces a whole system of elements by a pair of two components, where one component can be seen as a rotation and the other one as a translation. To alleviate the access to the mathematical background of Screw Theory and Lie Algebra, the basics or the often used terms are described.

- **Screw**

The Chasle theorem is commonly known: A spatial displacement can be depicted as a rotation and a translation parallel to the rotation axis or in line with it to describe the transformation of a rigid body movement from its first to its second position.

As rotation and the linear displacement are defined for one line, the image of a screw motion came up. In technical mechanics these six parameters to describe spatial displacement are characterized by three Euler Angles, which leads to the disadvantage of suffering singularities.

In contrary to the methods applied in common mechanics, the Screw Theory is an interesting upcoming instrument to facilitate the calculations. The screw itself is a pair of vectors, which consists of the four independent components of the Plucker vector that defines the screw axis [Wikipedia, ] on the one hand and the revolute angle about and the translational movement along this screw axis.

The outcome of this theory can be easily verified by imagining the rigid body under a constant spiral screw motion. By applying endless pitch to a screw the end route can be imagined as a straight path. The exact opposite is the pure rotation. It is represented if the screw motion has no pitch at all; that is the trajectory solely traces circles. Here it can be realized that the pitch of a screw relates their rotation and translational movement mutually.

The calculus and the transformation rules will not be described because it will go beyond the scope of this thesis. The interested reader detailed explanation in [Murray and Li, 1994].

- **Twist**

The infinitesimal version of a screw motion is called a twist [Murray and Li, 1994]. Screws are referred to as the screw of first order, while twists are also named as screws of second order. Equivalently they are also pairs of vectors but instead of containing the motion representation, twists offer the velocity description of a rigid body.

The first component is the angular velocity around the joint axis, the second one however is the linear velocity along the very axis.

$$\hat{v}_2 = \begin{bmatrix} \vec{v}_2 \\ \vec{\omega} \end{bmatrix} = \begin{bmatrix} 1 & -[\vec{r}_{12}]_{\times} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \vec{v}_1 \\ \vec{\omega} \end{bmatrix} \quad (2.1)$$

- $\vec{\omega}$  : Angular velocity of the rigid body.
- $\vec{v}_1$  : First point's linear velocity.
- $\vec{v}_2$  : Second point's linear velocity.
- $[\vec{r}_{12}]_{\times}$  : The  $3 \times 3$  cross product.

- **Wrench**

The last pairs of vectors in the line of screws are the wrenches. The first vector

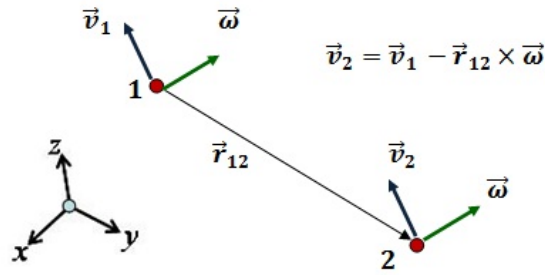


Figure 2.7: Twist: Pair of angular and linear velocity vector.

adhered to it is the force and the second one is the torque vector. Both have their origin in employing the Newton's laws to a rigid body. Congruently as explained in the paragraph of screws. Since forces depend on their point of application and their line of activity, they represent the Plucker coordinates of a spatial line with an infinite pitch. The torque is comparable to the rotary movement, it is a pure moment and independent of a line in space so it is a zero pitch screw.

$$\hat{t}_2 = \begin{bmatrix} \vec{F} \\ \vec{t}_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -[\vec{r}_{12}]_{\times} & 1 \end{bmatrix} \cdot \begin{bmatrix} \vec{F} \\ \vec{t}_1 \end{bmatrix} \quad (2.2)$$

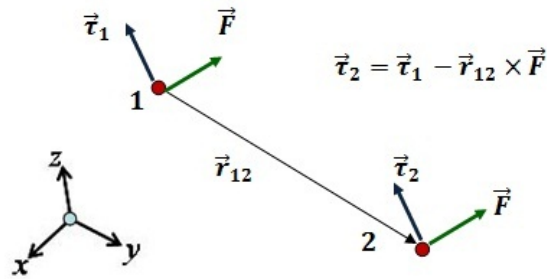


Figure 2.8: Wrench: Pair of force and torque vector.

- $\vec{\omega}$ : Resulting forces directed on the rigid body.
- $\vec{t}_1$ : First point's moment.
- $\vec{t}_2$ : Second point's moment.
- $[\vec{r}_{12}]_{\times}$ : The  $3 \times 3$  cross product.

To comprehend the automated model generation, which will be discussed in chapter "Implementation", these explanations may be pointed out first:

- **Homogeneous Representation:**

The transformation of points and vectors by rigid motion transformations have an uncomplicated representation regarding matrices and vectors  $\in \mathbb{R}^4$ . To achieve a vector  $\in \mathbb{R}^4$  the so called homogeneous coordinates, a 1 is appended to points coordinates. Considering vectors as difference of points, they have a 0 as their fourth coordinate.

- **Product of Exponentials:**

Instead of a complicated derivation of the joint angles based on given positions and orientations of frame, the product of exponentials of twists can be used to represents the kinematics of an open-chain mechanism.

- **Skew Symmetric Matrix:**

A Matrix  $M$  that satisfies the equation  $M = -M^T$  is a skewsymmetric one. In mathematics these matrices are often marked by a hat:  $\hat{M}$ .

For further examination of screw theory applied to robotics please refer to [Murray and Li, 1994] or [Siciliano and Khatib, 2008].

## 2.3 Geometric Formulation of Modular Robots

### 2.3.1 Connectivity Matrix

The *Connectivity Matrix*  $\mathcal{CM}(\vec{\mathcal{G}})$  is designating the accessibility between one module to its neighbour modules in a modular robot assembly. To obtain the Connectivity Matrix first a directed graph  $\vec{\mathcal{G}}$  of the kinematic chain needs to be generated. Modular robot assemblies can adopt a broad range of different appearances. Still every configuration can be depicted in a kinematic graph as it is a kinematic chain. The modular robot's kinematic graph needs to be in its directed form in order to derive its Connectivity Matrix.

In a kinematic graph every module is represented by a vertex and can be connected by edges, the joints of the module assemblies. A directed graph  $\vec{\mathcal{G}}$  differs from a ordinary graph  $\mathcal{G}$  since it delivers the directions of the connections. In a directed graph the direction of an edge points from the starting vertex to the reachable vertex. As commonly known a directed graph can be represented by a vertex-edge-matrix or a vertex-vertex-matrix, whose entries equal either one or zero. The Connectivity Matrix is represented by a vertex-vertex-matrix.

**Definition 2.1 (Connectivity Matrix)**

The *Connectivity Matrix* of a directed kinematic graph  $\vec{\mathcal{G}}$  of a modular robot with  $n + 1$  modules (vertices) is an  $(n + 1) \times (n + 1)$  matrix,  $\mathcal{CM}(\vec{\mathcal{G}}) = [cm_{ij}]$ ,  $i, j = 0, \dots, n$ , such that  $cm_{ij} = 1$ , if there is a directed path of length one solely from  $v_i$  to  $v_j$ ;  $cm_{ij} = 0$ , otherwise. [Chen and Yang, 1998]

Example of a Modular Robot:



Figure 2.9: Modular Robot Assembly with  $n = 9$  Modules and  $n - 1 = 8$  Joints.

$$\mathcal{CM}(\vec{\mathcal{G}}) = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (2.3)$$

The belonging Connectivity Matrix  $\mathcal{CM}(\vec{\mathcal{G}})$ .

**2.3.2 Accessibility Matrix**

The *Accessibility Matrix*  $\mathcal{R}(\vec{\mathcal{G}})$  is specifying the accessibilities between all the different modules in a modular robot assembly. To obtain the Accessibility Matrix first a directed graph  $\vec{\mathcal{G}}$  of the kinematic chain needs to be generated. The other way to achieve the Accessibility Matrix is to derive it from the Assembly Incidence Matrix by alternating the



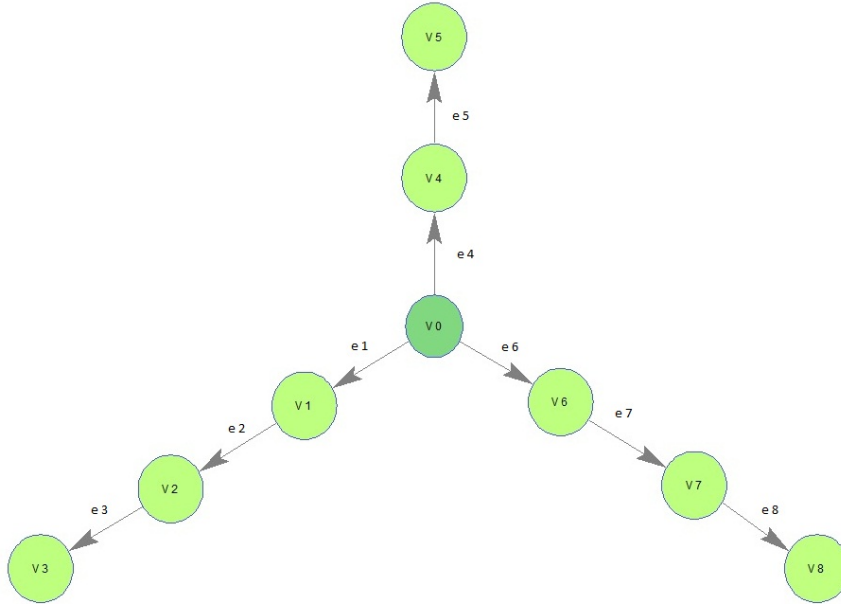


Figure 2.10: The Directed Kinematic Graph  $\vec{\mathcal{G}}$  with 9 Vertices  $v_i$  and 8 Edges  $e_j$  of the given robot.

### Connectivity Matrix.

The Connectivity Matrix and the Accessibility Matrix assume the same size consequently the Accessibility Matrix is also pictured in a vertex-vertex-matrix. Despite the fact that a modular robot can appear in various shapes the directed graph  $\vec{\mathcal{G}}$  and therefore the Accessibility Matrix of each peculiar modular robot is creatable to obtain a general representation form.

#### Definition 2.2 (Accessibility Matrix)

The *Accessibility Matrix* of a directed kinematic graph  $\vec{\mathcal{G}}$  of a modular robot with  $n + 1$  modules (vertices) is an  $(n + 1) \times (n + 1)$  matrix,  $\mathcal{R}(\vec{\mathcal{G}}) = [r_{ij}]$ ,  $i, j = 0, \dots, n$ , such that  $r_{ij} = 1$ , if there is a directed path of length one or more from  $v_i$  to  $v_j$ ;  $r_{ij} = 0$ , otherwise. [Chen and Yang, 1998]

If a modular robot incidence contains  $n$  modules, its joints are restricted to  $n - 1$  joints and the Accessibility Matrix is a  $(n) \times (n)$  matrix  $\mathcal{R}(\vec{\mathcal{G}}) = [r_{ij}]$ ,  $i, j = 0, \dots, n - 1$ . As there should not exist loops in a modular robot incidence all the diagonal entries of the Accessibility Matrix are set to zero to ensure there will be no directed edge from  $v_i$  to  $v_i$ .

By examining the Accessibility Matrix two main facts can be affirmed: The first information, the Accessibility Matrix reveals, is the quantity of paths. The number of paths

equals the number of all zero rows of the Accessibility Matrix. Secondly but not of minor importance the Accessibility Matrix shows from which vertices vertex  $v_i$  can be reached - the direction of the edge kept in mind. The Connectivity Matrix eases the derivation of the Accessibility Matrix. If  $cm_{ij} = 1$  row  $j$  has to be added to row  $i$ . This algorithm is to repeat for all the ones even the by adding newly generated ones. This computation accelerates the automatic model generation.

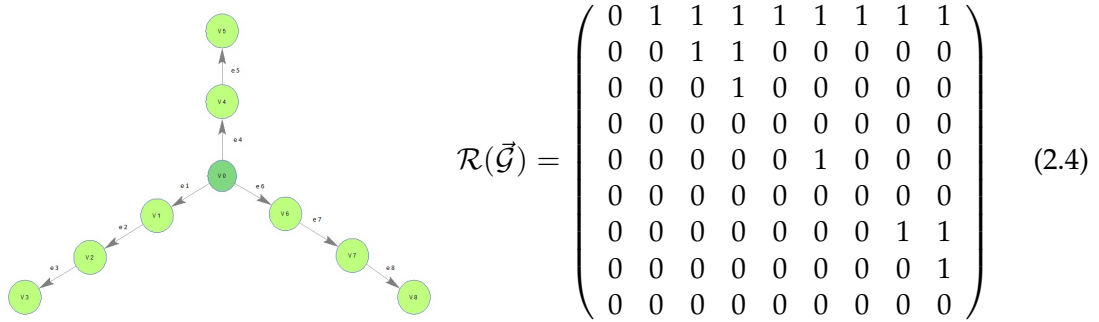


Figure 2.11: Example of the Modular Robot with  $n = 9$  Modules and  $n - 1 = 8$  Joints and the belonging Accessibility Matrix  $\mathcal{R}(\vec{\mathcal{G}})$ .

### 2.3.3 Assembly Incidence Matrix

The Assembly Incidence Matrix is an improved way to describe the modular robot assembly. In contrary to the Denavit-Hartenberg parameterization the Assembly Incidence Matrix characterizes both the module assembly sequence and the completed robots basic geometry. As it is explained in the passage of the Accessibility Matrix, a modular robot can be seen as a kinematic chain, which can be represented by a directed graph and as a result numerically in a vertex-edge incidence matrix, which is very important to derive an Assembly Incidence Matrix out of a directed graph or the other way round.

#### Vertex-Edge Matrix

The vertex-edge matrix  $\mathcal{M}(\vec{\mathcal{G}}) = [m_{ij}]$  is of a  $(n) \times (n - 1)$  form,  $i = 0, \dots, n - 1, j = 1, \dots, n - 1$ . This is obvious as a modular robot assembly consists of  $n$  link modules and connected by  $(n-1)$  joints.

**Definition 2.3 (Vertex-Edge Matrix)**

The *Vertex-Edge Matrix* of a directed kinematic graph  $\vec{\mathcal{G}}$  of a modular robot with  $n + 1$  modules (vertices) and  $n$  joints (edges) is an  $(n + 1) \times (n)$  matrix,  $\mathcal{M}(\vec{\mathcal{G}}) = [m_{ij}]$ ,  $i = 0, \dots, n$ ,  $j = 1, \dots, n$ , such that  $m_{ij} = 1$ , if there is a connection of  $v_i$  to  $e_j$ ;  $m_{ij} = 0$ , otherwise, [Chen and Yang, 1998].

The entry  $m_{ij}$  equals 1 if edge  $j$  has a connection to vertex  $i$ , otherwise the entry is set to zero. There is an easy way to check, whether the vertex-edge matrix is suitable or meets the requirements: Solely two entries of each column are allowed to equal 1, because one edge can only connect two ports or two modules respectively.

The Vertex-Edge Matrix  $\mathcal{M}(\vec{\mathcal{G}})$  can be converted into the Assembly Incidence Matrix  $\mathcal{A}(\vec{\mathcal{G}})$ .

**Definition 2.4 (Assembly Incidence Matrix)**

Let  $\vec{\mathcal{G}}$  be a kinematic graph of a modular robot and  $\mathcal{M}(\vec{\mathcal{G}})$  be its incidence matrix. Let  $port$  be the set of labels assigned to the connecting ports on the link modules. The *Assembly Incidence Matrix* of the robot  $\mathcal{A}(\vec{\mathcal{G}})$  is formed by substituting the 1s in  $\mathcal{M}(\vec{\mathcal{G}})$  with labels in  $port$  on respective modules. One extra column and row are augmented to  $\mathcal{A}(\vec{\mathcal{G}})$  to show the types of link and joint modules. [Chen and Yang, 1998]

Now to obtain the entries of the Assembly Incidence Matrix out of the vertex-edge matrix, all the ones are substituted by the labels of the connecting ports of each link.

Accordingly a method to distinguish the connecting ports of each link has to exist. There are various techniques to label these interfaces, thus labeling is not unique. But in this thesis only two methods are explained and the applicability of these two label techniques is restricted to cubic modules of all sizes.

The first one is a simple description according to the coordinates of the modules. Every single module annexes their middle to an imaginary Cartesian coordinate system. The six sides of the cube module can now be adjoined to the coordinate axes which cross their side orthogonally in positive or negative direction.

The second assignment follows the pattern of a gaming dice. According to the design of a gaming dice the sum of two opposite sides have to equal seven. By applying these techniques to label the sides of the cubes, the set of labels are:

$$L1 = \{-x, -y, -z, x, y, z\} \quad L2 = \{1, 2, 3, 4, 5, 6\} \quad (2.5)$$

Furthermore an additional row and an additional column are inserted to give consideration to both the geometry and the incidence sequence. The added row denotes whether the joints are prismatic, revolute, or virtual by the symbols  $Jp_i$ ,  $Jr_i$  and  $Jv_i$ . Respectively  $Lp_i$ ,  $Lr_i$ , and  $Lc_i$  are the denotations of prismatic, revolute, and cubic link modules. The index  $i$  explaining which joint or link is considered,  $i = n - 1$ . So the types of links and joints of a completed modular robot are indicated in the AIM. A special role plays the module  $v_0$  as it declares the base or root vertex and so is denoted by a  $B$ . In this thesis only cubic modules are worked with, which leads to the conclusion that denotations of link modules are abundant. Instead filling the entries of the last column with  $C_1$ ,  $C_2$  and so on to distinguish different sizes of cube modules is beneficial. In the example below, the color will differentiate between the cubes like visualized in Figure 2.9, so they are defined by  $C_p$ ,  $C_t$ ,  $C_b$  and  $C_g$  for pink, turquoise, black and green, respectively. A huge advantage of this representation method is that only a small component library is needed. This database preserves the specifications of the standard links and joint modules, sensors, actuators like the end-effectors. But it does not instruct the robots geometries or the sequences' types.

Different geometries of robot assemblies are conducted by calculating the sequences of the known link and joint modules. This raises the opportunity to gain a huge amount of information with a limited database or small background. All in all, the AIM together with the component database and the modeling methods for kinematics and dynamics are geometry-independent and thus the ideal way to automatically generate a model for modular robots.

The equivalent Vertex-Edge Matrix  $\mathcal{M}(\vec{\mathcal{G}})$  of the  $n = 9$  modular robot and its AIM  $\mathcal{A}(\vec{\mathcal{G}})$  with cube labels L2:

$$\mathcal{M}(\vec{\mathcal{G}}) = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad \mathcal{A}(\vec{\mathcal{G}}) = \begin{pmatrix} 4 & 0 & 0 & 3 & 0 & 5 & 0 & 0 \\ 5 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix} \quad (2.6)$$

The equivalent Assembly Incidence Matrix  $\mathcal{A}(\vec{\mathcal{G}})$  with coordinate labels L1 and the additional row and column:

$$\mathcal{A}(\vec{\mathcal{G}}) = \begin{matrix} & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 & \text{Links} \\ \begin{matrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \\ \text{Joints} \end{matrix} & \left( \begin{array}{cccccccc} -y & 0 & 0 & y & 0 & -x & 0 & 0 & B \\ -x & -y & 0 & 0 & 0 & 0 & 0 & 0 & C_p \\ 0 & y & x & 0 & 0 & 0 & 0 & 0 & C_t \\ 0 & 0 & -y & 0 & 0 & 0 & 0 & 0 & C_p \\ 0 & 0 & 0 & -y & y & 0 & 0 & 0 & C_b \\ 0 & 0 & 0 & 0 & -y & 0 & 0 & 0 & C_t \\ 0 & 0 & 0 & 0 & 0 & x & -y & 0 & C_b \\ 0 & 0 & 0 & 0 & 0 & 0 & y & -x & C_g \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & C_g \end{array} \right) & \end{matrix} \quad (2.7)$$

### 2.3.4 Path Matrix

The Accessibility Matrix helps to find unique ways from the base to the tips of tree-typed modular robots or in general to any link module. These unique ways are called a path. The Path Matrix considers solely the paths from the base to the very tips of every branch. This means that only those rows of the Accessibility Matrix are considered, which contain zeros in all entries. The ends of the paths are named pendant links and their quantity determines the number of paths.

#### Definition 2.5 (Path Matrix)

The *Path Matrix* of a directed kinematic graph  $\vec{\mathcal{G}}$  of a modular robot with  $n + 1$  modules (vertices) and  $p$  paths is an  $(p) \times (n + 1)$  matrix,  $\mathcal{P}(\vec{\mathcal{G}}) = [p_{ij}]$ ,  $i = 1, \dots, p$ ,  $j = 1, \dots, n$ , such that  $p_{ij} = 1$ , if path  $p_i$  contains vertex  $v_j$ ,  $p_{ij} = 0$ , otherwise. [Chen and Yang, 1999]

This means that the Path Matrix gives information about the quantity of paths, their lengths and sequences of modules and last but not least the pendant links.

The belonging Path Matrix  $\mathcal{PM}(\vec{\mathcal{G}})$ :

$$\mathcal{PM}(\vec{\mathcal{G}}) = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \quad (2.8)$$

The next chapter "Representation of Multi-Robot Organisms" will seize the general description and geometrical representation forms that were pointed out, engross the ideas and apply them to the present module types.

## 3 Representation of Multi-Robot Organisms

The chapter Representation of Multi-Robot Organisms is extending the chapter foundations. It comprises the adapted geometric formulation of modules due to different characteristics. Therefore it includes a brief analysis of the used hardware design of modular robot platforms backbone and scout, the specific modules of SYMBRION and REPLICATOR applied in this thesis. Moreover modification in representation involves alterations of kinematic and dynamical aspects of modular robots and thus implies further adjustment in the automatic model generation.

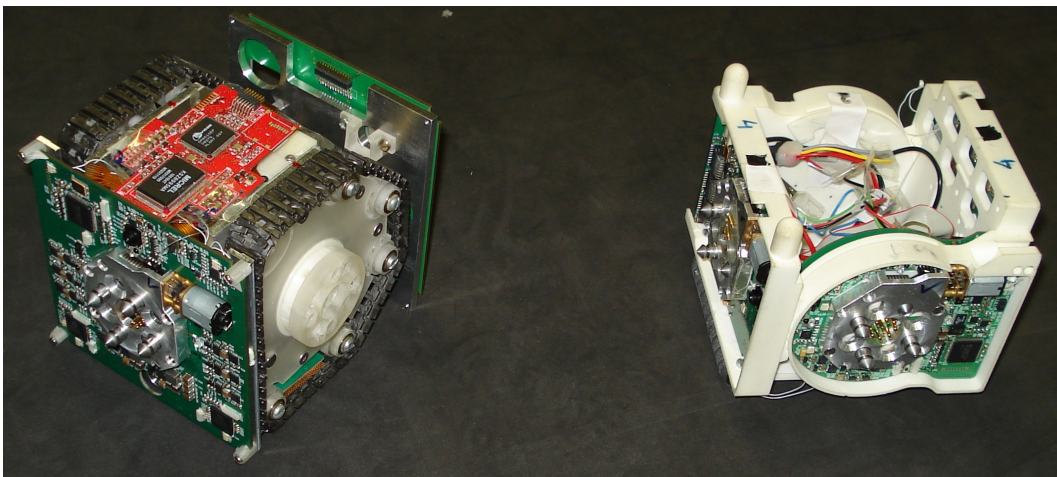


Figure 3.1: Scout and Backbone cube modules developed in SYMBRION and REPLICATOR

### 3.1 Adapted Module Design

The theoretician Chen and his companions defined their theory of automatic model generation for modular reconfigurable robot dynamics and kinematics to a main part based on the Assembly Incidence Matrix. This Assembly Incidence Matrix on the other hand is found on idealistic cube modules with perfect connectivity conditions. The modules are seen as homogeneous cubes with six identical sides and six connection ports. These connection ports are free to build up connection with other link modules in every of their six directions. This means that one module can interact with up to six other modules, which leads to innumerable connection possibilities as even three dimensional connections are

achievable.

Furthermore by applying Chen's method for this mentioned connections extra joint modules are expected. Extra joints refer to joint modules or connectors which are not integrated in the link module itself and can be attached to every arbitrary side. These joints can have three attainable movement possibilities. The first one is able to rotate around its joint axis and the second one allows a translational movement along its joint axis whereas the third one is a rigid joint and is not able to perform any movement at all.

But the reality offers a more complex basis of modules and joints, which differ from the named idealized cube shaped modules.

This Bachelor Thesis is introducing a concept of an Adapted Assembly Incidence Matrix to perform a similar calculation of the dynamics and kinematics of modular robot incidences based on modular robot platforms "Backbone" and "Scout". These two are heterogeneous state of the art platforms which are worked with and developed in various known projects as SYMBRION and REPLICATOR.

Due to growing interest and research in modular reconfigurable robots the platforms are already in the fifth generation but are still in an improving process. Especially in designing Printed Circuit Boards advancement can be achieved.

To obtain an impression of the analyzed platforms, "Backbone" and "Scout" cubes photographs are provided below.

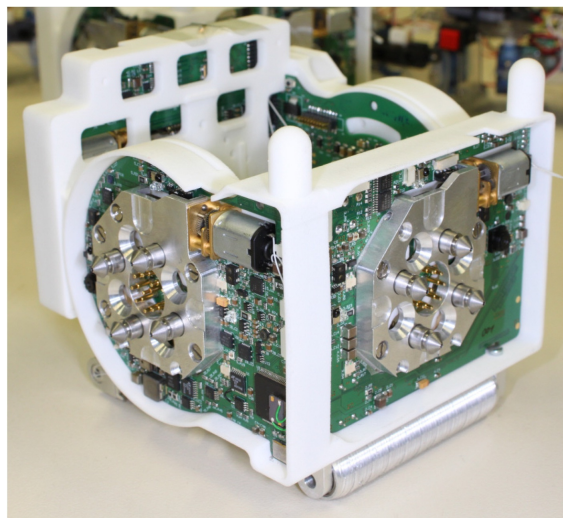


Figure 3.2: Backbone Platform: Front and Right Side. [Kernbach et al., 2011]

As one can see in Figure 3.1 and Figure 3.2 both the Scout and the Backbone platforms are manufactured slightly differently. Figure 3.2 proves that the "Backbone" platform is not a totally symmetric cube as requested from the cubes in theory. Only the front and the right side of the "Backbone" platform is pictured since the left and the right side are built equally. Thus the modules of the "Backbone" platform are symmetric to their diagonals



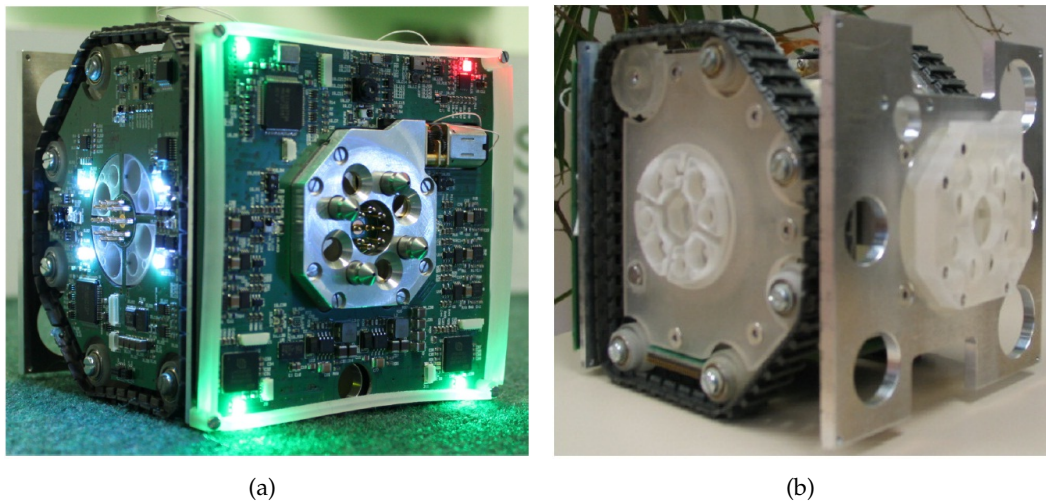


Figure 3.3: Scout Platform: (a) Front and Right Side. (b) Rear and Left Side (in Progress). [Kernbach et al., 2011]

as two congruent L-shaped elements are assembled to form a cube.

Figure 3.1 demonstrates that neither the "Scout" platform is a totally symmetric module as required in theory. In contrary to the "Backbone" platform, the "Scout" platform is not constructed of two equivalent elements but can be described by one U-shaped Element and an extra side.

But both platforms show the same main distinctness from the modules in the theoretical approach. The predominant difference is the joint module, which is already integrated in the link module. "Backbone" and "Scout" both have a rotary joint incorporated as one of their main components. The rotary axes are integrated in Figure 3.1

The cubes shown in Figure 3.1 are able to perform rotary movements around their joint axis. This circumrotation is proscribed to transgress the  $180^\circ$  limit.

Furthermore Figure 3.1 ensures that one module is in possession of only one Degree Of Freedom (DOF). To reassure this requirement of one DOF it is indispensable to connect two modules through a rigid connector, else the connector would add a second DOF. Two or more DOF should be avoided to keep the ordinary approximated structure of the Assembly Incidence Matrix to be able to preserve the general form of the automatic model generation. This leads to the conclusion that named two platforms are not allowed and not able to perform any translational movement at all.

Accordingly the distinction between joint and link modules is redundant. For that reason the adjustment of denomination has to be effected: In the following a module is designating a module with an integrated joint module unless it is explicitly called a joint or link module.



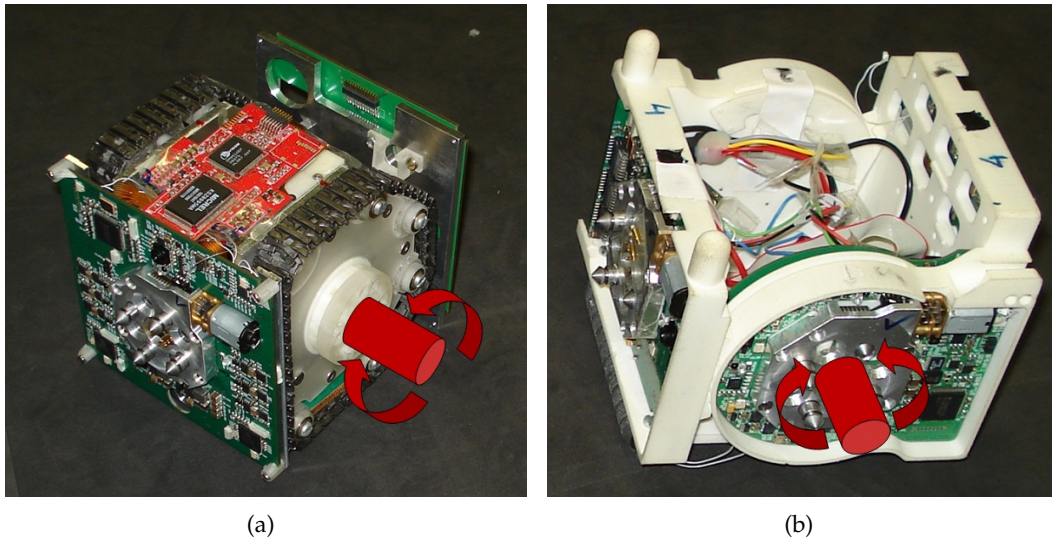


Figure 3.4: Integrated Rotary Joints: (a) Scout. (b) Backbone.

### 3.2 Adapted Assembly Incidence Matrix

Chen introduced the common Assembly Incidence Matrix as an essential part of their general method to generate a graph of a modular robot formation. The Assembly Incidence Matrix is a simple matrix that's purpose is to represent the robot assembly and its geometry. As the Assembly Incidence Matrix is depending on the current robot's constellation of its modules it cannot be independent of the features of the specific modules. If one consider a module of one of these platforms to be constructed like a gaming dice and having its top side labeled by one and the other surfaces congruently following the gaming dice pattern respectively, a design like in Figure 3.2 is the result.

The described alterations in module and joint description draw particular conclusions. The cubes shown in figures 3.2 are able to perform rotary movements around its joint axis. This circumrotation is proscribed to transgress the  $180^\circ$  limit, which allows the connection side, currently labeled by three, to obtain additionally the positions held by the sides labeled with one and six.

As discussed before the modules in Figure 3.2 are able to perform rotary movements around its joint axis limited to a total of  $180^\circ$  by a rotation maximum of  $+90^\circ$  or  $-90^\circ$ .

The outcome of this  $180^\circ$  rotation of the third surface is that sides one and six do not exist as proper PCB sides with connection ports to enter a relationship with another module. Hence sides one and six should be seen as placeholders rather than independent sides and are marked as red PCBs.

To simplify the assembling of two robot modules the following assumption is made:

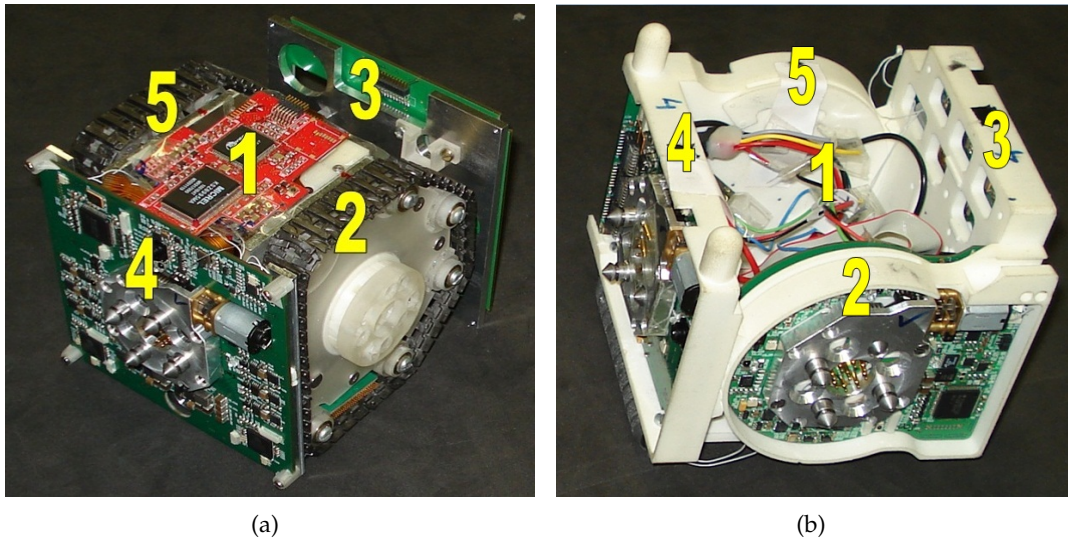


Figure 3.5: Modules Labeled by L2: (a) Scout. (b) Backbone.

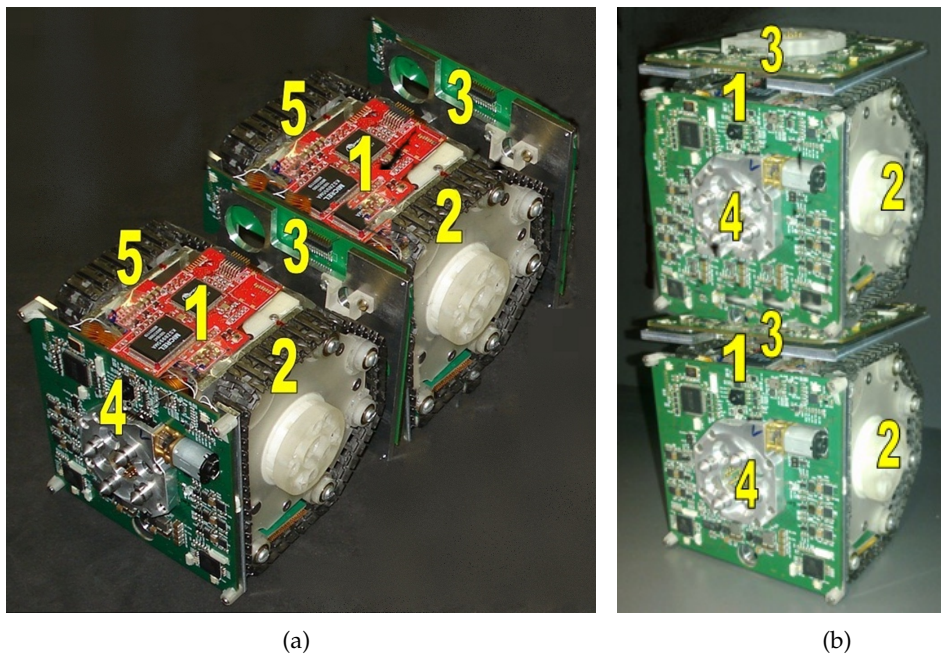


Figure 3.6: Scout Platform: (a) Dyad Labeled with Gaming Dice Pattern L2. (b) Side three rotated by  $90^\circ$ .

**Theorem 3.2.1 (Planar Connection)**

Modular robots do only enter a connection in the horizontal plane. Under no conditions a connection leading to a third dimension is to be established.

This raises the first restriction: It is obvious that only a connection between the sides two, three, four or five can be built up and held.

A second constraint has to be acknowledged due to the varying connection ports of the two platforms. As mentioned above the theoretical approach is based on equivalent connection ports which establish their mechanical, power and control connection. These platforms though force us to characterize and differentiate between two versions of connection ports. On the one hand exists an active and on the other hand a passive connector. The former docking unit is described as CoBoLD, Cone Bolt Locking Device, which owned its name from its shape. As it is genderless it is able to establish a connection with all the other docking elements, whether they are passive or not. The passive locking devices are also genderless but are only able to connect with an active element. The connection itself regarding power and control must not be distinguished. Further information in [Kernbach et al., 2011].

As it is very likely that the platforms soon in the near future will be constituted of only active connecting ports, all generally possible connections regarding the platforms will be included in the calculations and the second restriction will be neglected in the following.

Nonetheless a modification of the normal Assembly Incidence Matrix is absolutely necessary. An Assembly Incidence Matrix teaches the contemplator to evaluate the geometry and the sequence of modules of the specified robot assembly matrix. Due to the following reasons the Adapted Assembly Incidence Matrix will be introduced later on if all the alterations are explained:

- **Vertex-Edge:** The vertex-edge matrix cannot be seen as equivalent to the one specified in the anterior section. On the one hand it offers the same information for explaining which module is connected to which module but on the other hand it does not represent the joints as there are no such extra joint objects as specified by Chen.
- **Labels:** As far as the labels are concerned, the gaming dice pattern will be used with the exclusion of the numbers one and six since these are the forbidden connecting ports to ensure that only planar connections will be established.
- **Additional Column:** Moreover the cube modules do suffer a change in definition. The cubes are seen to be symmetric regarding the normal to the joint axis which passes almost the middles of sides two and five. The additional column, which describes the module type, does not convey further information as all modules are of the same size. The exact measurement can be seen in the section, which specifies the two platforms.

- **Additional Row:** Even the additional row, which describes the joint types in Chen's Assembly Incidence Matrix, has to cope with alterations as a translational movement between two link modules is not possible at all using the considered platforms leaving only the rigid or revolute joint as an option. As every module contains a revolute joint in its center but establishes a rigid connection with another module to form a dyad, the additional row as it is stated by Chen does not impart extra knowledge.

However various configurations between modules are possible to create dyads. On the one hand the difficulty in obtaining the Adapted Assembly Incidence Matrix is located in the different movements which can be obtained due to different assemblies of sides. On the other hand the difficulty primarily is caused by the lack of joints as extra joint modules. Before introducing the Adapted Assembly Incidence Matrix a few foundations will be explained below.

Consequently the section "Joint Axes Configurations" explains the basic axes constellation of one dyad and an auxiliary look-up-table will be introduced to separate the individual appearances of the dyads easily.

### 3.2.1 Joint Axes Configurations

All in all there exist two main movements next to the rigid connection. The movement are caused by the integrated rotating joints and the varying orientation of their axes. For a caterpillar-like movement the joint axes of the dyads need to be in line. If they are orthogonally collocated, movement in rectangular directions is possible. For example a body torso with arms and legs can be implemented.

As announced earlier this Bachelor Thesis provides a look-up table, which allows an easy way to differentiate between the orthogonal and the parallel connection of two modules in a dyad with one look. As described above joint axes influence the movement patterns and therefore have to be divided into two main classes, the generally serial and the orthogonal axes, which will be denoted by the set  $\mathcal{S} \dots$  and set  $\mathcal{SO}$  respectively.

Furthermore a finer classification of the former group is necessary as the serial connection of a dyad, can end up with parallel axes or with all joint axes situated on one line. This is why set  $\mathcal{S} \dots$  is will be labeled by  $\mathcal{SS}$  if all the axes are in line and otherwise serial modules which axes are parallel arranged are denoted by  $\mathcal{SP}$ .

The first letter  $\mathcal{S}$  of all three groups is implying the serial combination of two modules. All groups have that feature in common as only dyads are considered. The second letter indicates the mutual constellation of the axes and can presume the letters  $\mathcal{S}$ ,  $\mathcal{O}$ ,  $\mathcal{P}$  and specifies the serial, orthogonal and parallel axes.

In the following label two and five are seen as elements of the  $\mathcal{SS}$  - set:  $2, 5 \in \mathcal{SS}$  or  $\mathcal{SS} = \{2, 5\}$  and the elements three and four are members of the  $\mathcal{SP}$ -set:  $3, 4 \in \mathcal{SP}$  or  $\mathcal{SP} =$

$\{3, 4\}$ .

The general group  $\mathcal{GG}$  of possible link connection variations is:

$$\begin{aligned} \mathcal{GG} = \{ & (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (2, 1), (2, 2), (2, 3), \\ & (2, 4), (2, 5), (2, 6), (3, 1), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), \\ & (4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (5, 1), (5, 2), (5, 3), \\ & (5, 4), (5, 5), (5, 6), (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6) \} \end{aligned} \quad (3.1)$$

The state of the art development of the link cubes with by design limited mechanical conditions induce the following constraints  $\mathcal{C}$ ,  $\mathcal{CSS}$ ,  $\mathcal{CSP}$  and  $\mathcal{CSO}$  which have to be subtracted from  $\mathcal{GG}$ , to obtain the designated set:

The lack of connection ports on side one and six, which forces the link modules to connect only if they are in their horizontal plane form, when standard configurations are achieved, is formulated by the general constraint  $\mathcal{C}$ , which is principally valid and has to be applied to all connections:

$$\begin{aligned} \mathcal{C} = \{ & (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), \\ & (2, 1), (2, 6), (3, 1), (3, 6), (4, 1), (4, 6), \\ & (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6), \\ & (5, 1), (5, 6) \} \end{aligned} \quad (3.2)$$

It implies that neither connection port one nor connection port six enters a connection with another module.

To guarantee that all axes are parts of one line, constraint  $\mathcal{CSS}$  is compulsory:

$$\begin{aligned} \mathcal{CSS} = \{ & (2, 3), (2, 4), (3, 2), (3, 3), (3, 4), (3, 5), \\ & (4, 2), (4, 3), (4, 4), (4, 5), (5, 3), (5, 4) \} \end{aligned} \quad (3.3)$$

$\mathcal{CSP}$  confirms that all axes are parallel:

$$\begin{aligned} \mathcal{CSP} = \{ & (2, 2), (2, 3), (2, 4), (2, 5), (3, 2), (3, 5), \\ & (4, 2), (4, 5), (5, 2), (5, 3), (5, 4), (5, 5) \} \end{aligned} \quad (3.4)$$

Last but not least  $\mathcal{CSO}$  verifies the orthogonality of all the combinations of modules:

$$\begin{aligned} \mathcal{CSO} = \{ & (2, 2), (2, 5), (3, 3), (3, 4), (4, 3), (4, 4), \\ & (5, 2), (5, 5) \} \end{aligned} \quad (3.5)$$

Then the assignment of a connection variety to a dyad, the pair of two combined link modules, is easy. To obtain set  $\mathcal{SO}$ , for orthogonal joint axes additional constraint  $\mathcal{CSO}$  has to be included in  $\mathcal{C}$ :

$$\begin{aligned} \mathcal{SO} &= \mathcal{GG} \setminus \mathcal{C} \cup \mathcal{CSO} \\ &= \{(2,2), (2,3), (2,4), (2,5), (3,2), (3,5), \\ &\quad (4,2), (4,5), (5,2), (5,3), (5,4), (5,5)\} \end{aligned} \quad (3.6)$$

The set  $\mathcal{SS}$  is given by reducing  $\mathcal{GG}$  by set  $\mathcal{SO}$  and the additional constraint  $\mathcal{CSS}$ :

$$\begin{aligned} \mathcal{SO} &= \mathcal{GG} \setminus (\mathcal{C} \cup \mathcal{CSS} \cup \mathcal{SO}) \\ &= \{(2,2), (2,5), (5,2), (5,5)\} \end{aligned} \quad (3.7)$$

$\mathcal{SP}$  can be calculated by subtracting  $\mathcal{CSP}$  and  $\mathcal{SO}$  from  $\mathcal{GG}$

$$\begin{aligned} \mathcal{SP} &= \mathcal{GG} \setminus (\mathcal{C} \cup \mathcal{CSP} \cup \mathcal{SO}) \\ &= \{(3,3), (3,4), (4,3), (4,4)\} \end{aligned} \quad (3.8)$$

Table 3.1 gives an overview about the different serial combination possibilities to distinguish between the individual appearances of the dyads easily:

Set SS: Serial Modules, Serial Axes		Set SP: Serial Modules, Parallel Axes		Set SO: Serial Modules, Orthogonal Axes	
1 <sup>st</sup> Mod.	2 <sup>nd</sup> Mod.	1 <sup>st</sup> Mod.	2 <sup>nd</sup> Mod.	1 <sup>st</sup> Mod.	2 <sup>nd</sup> Mod.
2	2	3	3	2	3
2	5	3	4	2	4
5	2	4	3	3	2
5	5	4	4	3	5
				4	2
				4	5
				5	3
				5	4

Table 3.1: Look-Up Table for Dyads labeled by  $\mathcal{L} \in$  and their Axes Constellations.

The look-up-table 3.1 permits three general axes constellations of two link modules.



To point out the differences between the three connection varieties, examples of dyads as members of each set are presented, offering the standard configuration and the spectrum of movement possibilities. The following figures show models of the platform type "Scout". In order to keep the figures standardized green illustrates theoretically enabled connecting ports whereas red is highlighting the forbidden ones.

Light green indicates the punctures of the rotation axes. This means that the cube sides two and five are always painted in light green. The joint axes are furthermore marked by a bidirectional arrow clarifying that circulation is caused by these axes and that these are the lines around which particular sides can revolve.

Dark green is reserved for sides three and four.

Red is the color for inhibited connection. As side one and its opposing side six are not able to enter any connection due to prohibited three dimensional connections, they are colored red.

If a rotation took place the faded side three will be marked orange, because the connection port wandered from side three or four to side one or six or the other way round. The orange colored labels can theoretically enter a relationship with another module if the whole assembly assumes the two-dimensional shape and the rotating elements their initial position, the standard configuration.

These rules reduce the complexity of robot assembly figures and show details about executable revolving deformation and accomplishable geometries.

The DOF of these dyads which models the movement pattern under a joint displacement are illustrated in the next paragraph.

- Set  $S\mathcal{O}$ :

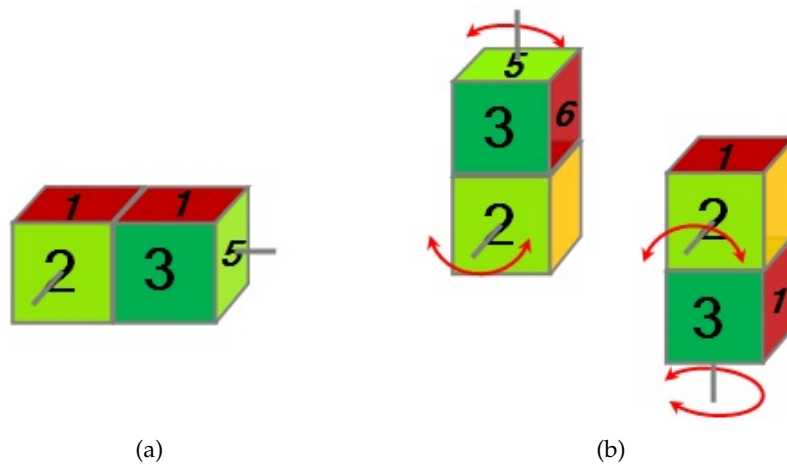


Figure 3.7: Dyad Configurations  $S\mathcal{O}$ : (a) Type (3,4)  $S\mathcal{O}$  Standard Configuration (b) Rotation Movement limited to a total of  $180^\circ$ .

• Set  $\mathcal{SP}$ :

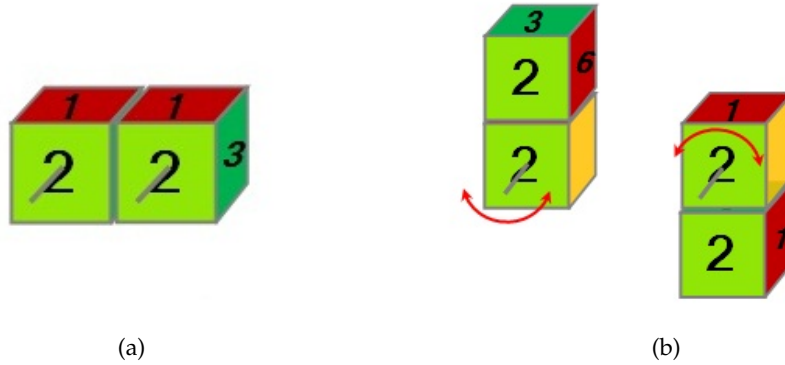


Figure 3.8: Dyad Configurations  $\mathcal{SP}$ : (a) Type (3,4)  $\mathcal{SP}$  Standard Configuration (b) Rotation Movement limited to a total of  $180^\circ$ .

• Set  $\mathcal{SS}$ :

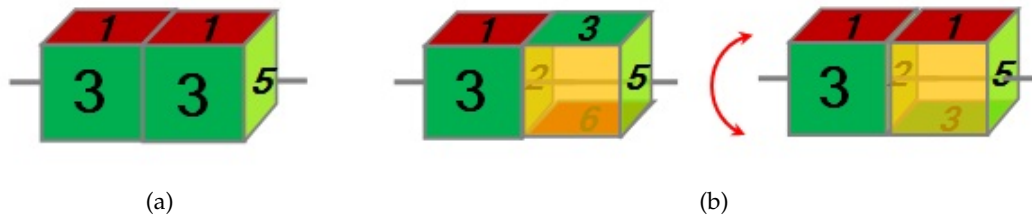


Figure 3.9: Dyad Configurations  $\mathcal{SS}$  focusing on side three: (a) Type (5,2)  $\mathcal{SS}$  Standard Configuration (b) Rotation Movement limited to a total of  $180^\circ$ .

Absolutely important is the fact that not the module itself is revolving around the axis, what would require a joint between the two links, but the side three solely. This guarantees an independent movement of the second link module relative to the first link module. Of course side three of the first link module also might perform a revolte movement of  $180^\circ$ , which is not shown in another graphic. A difficulty though occurs if side four performs a rotation, if the platform "Scout" is used as it is U-shaped. It forces sides two and five to adapt exactly to and further to execute its movement. A second example of  $\mathcal{SS}$  3.2.1 demonstrates this uniqueness.

Obviously independent movement of the two elements of the dyad is not possible if a rotation of side four is imposed on it. Any rotation of side four in this constellation enforces the adjacent link to join the behavior. There is a rigid connection between the link modules in both constellation, but the differences is occulted in the rigid connection between sides two, four and five due to the described U-shape. Only



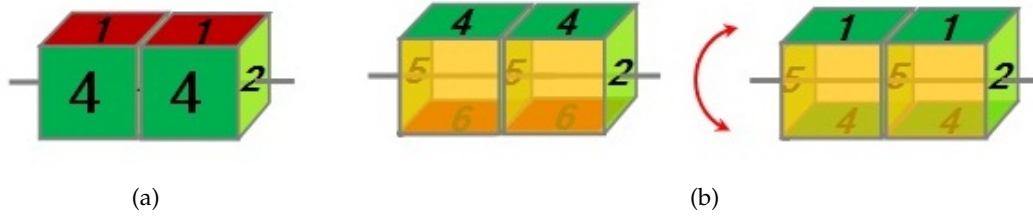


Figure 3.10: Dyad Configurations  $SS$  focusing on side four: (a) Type (2,5)  $SS$  Standard Configuration (b) Rotation Movement limited to a total of  $180^\circ$ .

side three is a completely independent side regarding its moving sector. These two patterns are all possible constellations of the  $SS$  group.

After explaining the joint configuration sets, the definition of the AAIM can be presented.

**Definition 3.1 (Adapted Assembly Incidence Matrix)**

Let  $\vec{\mathcal{G}}$  be a kinematic graph of a modular robot and  $\mathcal{M}(\vec{\mathcal{G}})$  be its incidence matrix. Let  $port$  be the set of labels assigned to the connecting ports on the link modules. The *Adapted Assembly Incidence Matrix* of the robot  $\mathcal{AAIM}(\vec{\mathcal{G}})$  is formed by substituting the 1s in  $\mathcal{M}(\vec{\mathcal{G}})$  with labels in  $port$  on respective modules. One extra column and row are augmented to  $\mathcal{AAIM}(\vec{\mathcal{G}})$  to show the types of link modules and joint connection varieties. The *Adapted Assembly Incidence Matrix* of a directed kinematic graph  $\vec{\mathcal{G}}$  of a modular robot with  $n + 1$  modules (vertices) and  $n$  joints (edges) is an  $(n + 2) \times (n + 1)$  matrix,  $\mathcal{AAIM}(\vec{\mathcal{G}}) = [aaim_{ij}]$ .

The AAIM differs from the AIM as follows. The last row recites the joint configuration set as the mentioned joints do not exist as extra joint modules but are the connection sides of the first link module. The last column solely declares the basis  $b$ .

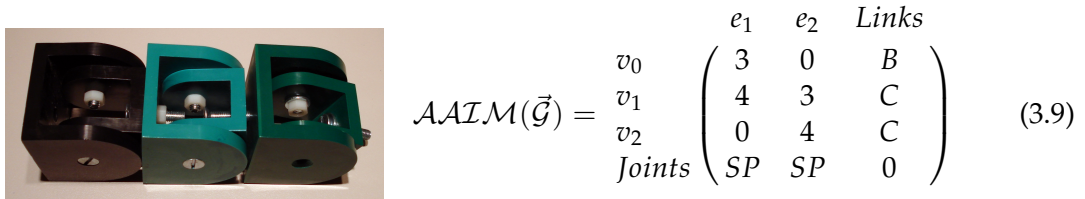


Figure 3.11: A serial typed robot consisting of  $n = 3$  modules and its  $\mathcal{AAIM}$ .

## 4 Implementation

The chapter Implementation consists of two main components. Firstly it provides a general overview of the automated model generation for modular reconfigurable robot dynamics and secondly it explains the framework developed during this thesis step by step. A few fragments of the MATLAB Code, which produces the geometric representation of the matrices and partial extracts of the Forwards Kinematics are integrated in the explanations.

### 4.1 Automatic Model Generation

In the former chapters "Foundations" and "Representation of Multi-Robot Organisms" the Adapted Assembly Incidence and the kinematic models of dyads were discussed in detail. Together they form the prerequisites to initiate the forward kinematics calculations of branched or serial typed modular robot with no closed loops.

To understand the calculations of the entire modular robot, they are developed out of the recursive kinematics calculations of single dyads according to the Depth-First-Search algorithm, the graph traversing order, which has to be determined in advance.

#### 4.1.1 Dyad Kinematics

As described in Foundations two connected modules are seen as a dyad, where the first link module is denoted by  $v_i$  and the adjacent one by  $v_j$ . Basically the connecting joint module is referred to as  $e_j$ . The indices point out that  $v_j$  and  $e_j$  are congregated and called link assembly  $j$  in the following.

The indices of all matrices, calculations or transformations  $E$  indicate in which frames they are expressed:

- $E_i$ :  $E$  is expressed in frame  $i$ .
- $E_j$ :  $E$  is expressed in frame  $j$ .
- $E_{ij}$ :  $E$  is expressed in frame  $j$  relative to frame  $i$ .

The order, in which the modules are denoted, is determined by the graph traversing algorithm. In the following the Depth-First-Search is employed. It declares module  $v_0$  as base module  $b$  and its neighbor module as  $v_1$ .

$e_0$  is not specified in the Assembly Incidence Matrix as it is the additional joint but will be considered in the calculation by the joint angle  $q_0$ .

Both the frames and the additional basis joint are visualized in Figure 4.1.1:

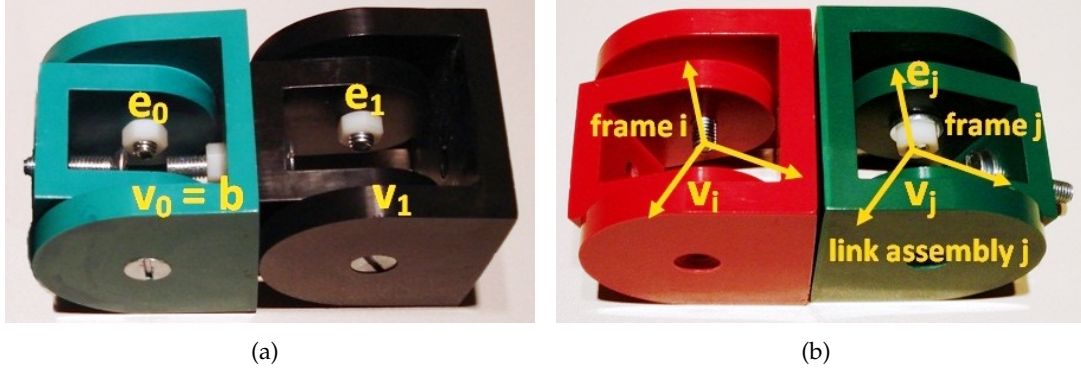


Figure 4.1: Dyad Kinematics: (a) Basis b: Additional Joint Module  $e_0$  of Link Module  $v_0$  .(b) Visualization of Dyads' Denotations and Frames.

The relative pose  $T_{ij}$  of module  $j$  relative to module  $j$  can be represented in a  $4 \times 4$  homogeneous matrix. In pose both the position and orientation are included.

$$T_{ij}(q_j) = T_{ij}(0) \exp \hat{s}_j q_j \quad (4.1)$$

$$T_{ij}(0) = \begin{bmatrix} R_{ij}(0) & d_{ij}(0) \\ 0 & 1 \end{bmatrix} \quad (4.2)$$

$$\hat{s}_j = \begin{bmatrix} \hat{w}_j & v_j \\ 0 & 0 \end{bmatrix} \quad (4.3)$$

- $T_{ij}$ :  
The  $4 \times 4$  homogeneous  $T_{ij}$  describes relative pose of frame  $j$  relative to frame  $i$ . Both the relative and the following initial pose are members of the Euclidean group  $SE(3)$ .
- $T_{ij}(0)$ :  
Like  $T_{ij}$  the initial pose  $T_{ij}(0)$  of frame  $j$  relative to frame  $i$  is also a  $4 \times 4$  homogeneous matrix.
- $R_{ij}(0)$ :  
As  $T_{ij}(0)$  contains the initial position and the initial orientation, the matrix  $R_{ij}(0)$  represents the orientation whereas
- $d_{ij}(0)$ :  
The  $3 \times 1$  vector  $d_{ij}(0)$  tells the initial position of frame  $j$  relative to frame  $i$ .
- $\hat{s}_j$ :  
The Matrix  $\hat{s}_j$  represents the twist of joint  $e_j$  and is furthermore element of  $se(3)$  the Lie Algebra of  $SE(3)$ . As described in the screw theory section,  $\hat{s}_j$  is the skew-symmetric representation in matrix form of the six elements row vector  $s_j = (v_j, w_j)$ .

- $v_j$ :  
The three elements line vector  $v_j$  represents the translational part of the relative positions' distance. If  $p_j$  is a position vector of any point along the joint axis, then it is defined as follows:  $v_j = (v_j, v_j, v_j) = w_j \times p_j$ .
- $w_j$ :  
The three elements row vector  $w_j$  depicts the normalized directional vector of joint  $j$  relative to its frame. Therefore it is of immense importance for the revolute joints as  $w_j$  describes the relative rotation.
- $q_j$ :  
Joint angles of link module  $j$  will be denoted by  $q_j$ .

The explanations of  $\hat{s}_j$ ,  $w_j$  and  $v_j$  point out that for a pure rotation equation  $s_j = (v_j, w_j)$  assumes the reduced form:  $s_j = (0, w_j)$ , in contrary to prismatic joints which produce this simplification:  $s_j = (0, w_j)$ .

$$\begin{aligned} T_{0n}(q) &= T_{01}(q_1)T_{12}(q_2)T_{23}(q_3) \dots T_{n-1n}(q_n) \\ &= T_{01}(0) \exp \hat{s}_1 q_1 T_{12}(2) \exp \hat{s}_2 q_2 T_{23}(0) \exp \hat{s}_3 q_3 \dots T_{n-1n}(0) \exp \hat{s}_n q_n \end{aligned} \quad (4.4)$$

MATLAB-Code:

```
% Dyad Frame Configuration
% T_ij = 4x4 homogeneous Matrix
% Determine dij and Rij for all Links in each Path
% Tij0cell{x,y}= initial pose of Frame y relative to x
Tij = sym(eye(4));
T_0 = eye(4);
Tij0cell = cell(max(pos(:,1)),max(pos(:,2)));
for x = 1:r
    for y = 1:r
        Tij0cell{x,y} = eye(4);
    end
end

for x = 1:c
    if (pairs(x,1) == 2)
        dij = [1 0 0]';
    elseif (pairs(x,1) == 3)
        dij = [0 1 0]';
    elseif (pairs(x,1) == 4)
        dij = [0 -1 0]';
    elseif (pairs(x,1) == 5)
        dij = [-1 0 0]';
    end
end
```

```

else
    disp('Error occured');
end
if ((lastrowAAIM(x)~= 0) && (pairs(x,1) + pairs(x,2) ~= 7))
    Rij = [-1 0 0;
           0 -1 0;
           0 0 1];
else
    if (( pairs(x,1)== 2 && pairs(x,2) == 3)...
        ||(pairs(x,1)== 3 && pairs(x,2) == 5)...
        ||(pairs(x,1)== 4 && pairs(x,2) == 2)...
        ||(pairs(x,1)== 5 && pairs(x,2) == 4))
        Rij = [0 -1 0;
               1 0 0;
               0 0 1];
    else
        Rij = [0 1 0;
               -1 0 0;
               0 0 1];
    end
    Tij(1:3,4) = vpa(dij);
    Tij(1:3,1:3) = vpa(Rij);
    Tij0cell{pos(x,1),pos(x,2)} = Tij;
end
end
Tij0mat = [T_0;cat(1,Tij0cell{:})];

```

With this dyad kinematics knowledge in mind the forward kinematics for a whole modular robot can be calculated in a recursive way.

### 4.1.2 Forward Kinematics

[Chen and Yang, 1996] proposed a straightforward formula to calculate all the forward transformations of a modular branching robot recursively. For that reason the so called TreeRobotKinematics Algorithm calculates the forward transformations of a modular robot starting from its base link to its entire pendant links according to the pattern represented in Figure 4.2.

In the following, as well as in the framework, the Depth-First-Search algorithm is employed because the traversing order is obviously easier to recognize than the Breadth-First-Search-Algorithm, despite the fact that both be similiar regarding their complexity.

To start this algorithm three points have to be clear. Firstly the Adapted Assembly In-

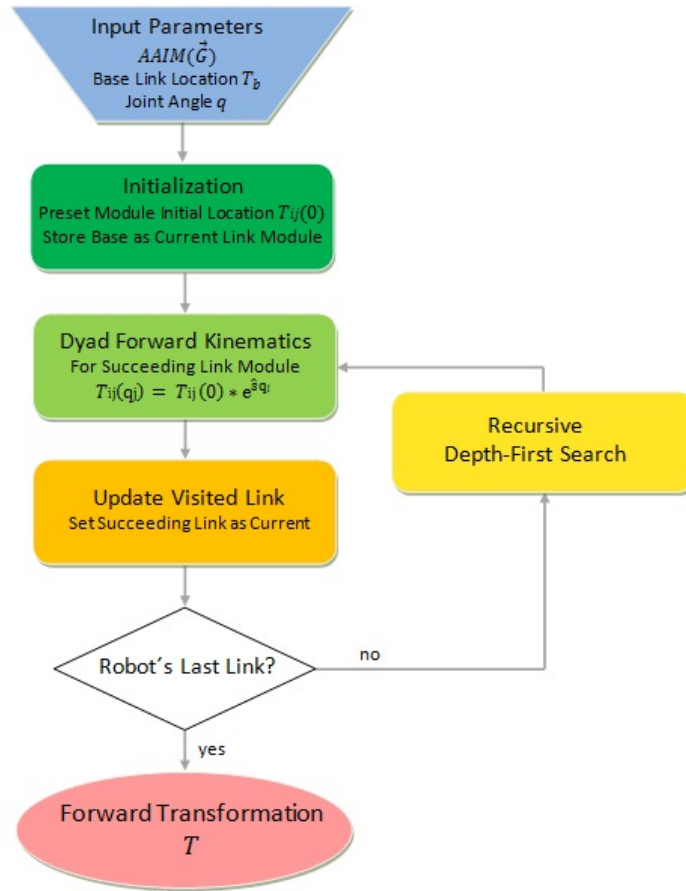


Figure 4.2: TreeRobotKinematics Algorithm.

cidence Matrix  $AAIM(\vec{G})$ , to determine the assembly's geometry, secondly the pose of the base module  $T_b$  and lastly all the angles configurations  $q$  of all the joints including  $e_0$  due to the used platforms. Then the default value will be assigned to initial module location  $T_{ij}(0)$  and the algorithm starts with the base module set as the current link module.

Afterwards a loop to develop the poses of all links, the dyads forwards kinematics is passed through until the last link module of the robot according to the Depth-First-Search algorithm is found. At the end all the forward transformations, namely all the positions and orientations of all link modules in relation to the base frame are determined.

### 4.1.3 Dynamics

This subsection is dedicated to achieve an accurate dynamic model of a branching type modular robot. Because the Newton-Euler equation and the Lagrangian Method in its closed-form have an equivalent relationship, the implementation of the following illus-

tration delivers the equations of motions of any modular robot assembly and therefore its Dynamic Model.

Although normally it is indispensable to calculate a transformation matrix to ensure that the mass frame  $j^*$  of the link assembly and the ordinary frame  $j$  of the link module coincide with each other, in the case of the applied platforms the transformation can be neglected as the joint is already incorporated.

This is why the Newton-Euler equation of link assembly  $j$  can directly be written in the coordinate frame  $j$  of its link module without applying the Huygens-Steiner theorem.

Please note, that the Newton-Euler equation is transformed into its adjoint representation.

$$F_j = M_j \dot{V}_j - ad_{V_j}^T(M_j V_j) \quad (4.5)$$

$$M_j = \begin{bmatrix} m_j I & m_j \hat{p}_{j^*j} \\ m_j \hat{p}_{j^*j} & J_{j^*} - m_j \hat{p}_{j^*j}^2 \end{bmatrix} \quad (4.6)$$

- $F_j$ :  
The  $6 \times 1$  column vector  $F_j = [f_j, \tau_j]$  consists of the resultant forces  $f_j$  and all the applied torques  $\tau_j$ , both  $\in \mathbb{R}^{3 \times 1}$ . Thus it describes the resultant wrench applied to the link assembly  $j$ .
- $M_j$ :  
Represents the Generalized Mass Matrix  $M_j \in \mathbb{R}^{6 \times 6}$  under the condition that the coordinate axes of the frame  $j$  and the mass frame  $j^*$  parallel.
- $V_j$ :  
The generalized body velocity, a  $6 \times 1$  column vector  $V_j = [v_j, w_j]$ . It comprises body translational velocity  $v_j$  as well as  $w_j$ , the angular velocity of the body.
- $\dot{V}_j$ :  
Like the generalized body velocity  $V_j$ , the generalized body acceleration  $\dot{V}_j$  is a  $6 \times 1$  column vector.
- $ad_{V_j}^T$ :  
The transposed version of the adjoint matrix of  $V_j$  is  $ad_{V_j}^T \in \mathbb{R}^{6 \times 6}$ .
- $\hat{p}_{j^*j}$ :  
Is declared as the skew-symmetric matrix representation of  $p_{j^*j}$ , the relative position vector of frame  $j$  and  $j^*$ .

$$ad_{V_j}^T = (ad_{V_j})^T = \begin{bmatrix} \hat{w}_j & \hat{v}_j \\ 0 & \hat{w}_j \end{bmatrix}^T = \begin{bmatrix} -\hat{w}_j & 0 \\ -\hat{v}_j & -\hat{w}_j \end{bmatrix} \quad (4.7)$$

$$Ad_{T_{mn}}^T = (Ad_{T_{mn}})^T = \begin{bmatrix} R_{mn} & \hat{p}_{mn} R_{mn} \\ 0 & R_{mn} \end{bmatrix}^T = \begin{bmatrix} R_{mn}^T & 0 \\ -R_{mn}^T p_{mn} & R_{mn}^T \end{bmatrix} \quad (4.8)$$

- $ad_{V_j}^T$ :  
The transposed version of the adjoint matrix of  $V_j$  is  $ad_{V_j}^T \in R^{6 \times 6}$ .
- $Ad_{T_{mn}}^T$ :  
This is the transpose of the adjoint presentation of  $T_{mn} \in SE(3)$  it is used to transform forces, torques, velocities and acceleration from frame  $m$  to frame  $n$ .

### Recursive Newton-Euler Algorithm

The recursive Newton-Euler Algorithm can be subdivided into two main processes, the forward and the backward iteration. This is why it is called a two-step iteration process. The forward iteration produces the generalized velocity and the generalized acceleration from the base module to the ends of all branches of a robot assembly.

In contrast to the forward iteration the generalized forces of all branches are propagated backwards from the tips to the root module in the backward iteration.

$$Ad_{T_{mn}}^T = (Ad_{T_{mn}})^T = \begin{bmatrix} R_{mn} & \hat{p}_{mn}R_{mn} \\ 0 & R_{mn} \end{bmatrix}^T = \begin{bmatrix} R_{mn}^T & 0 \\ -R_{mn}^T p_{mn} & R_{mn}^T \end{bmatrix} \quad (4.9)$$

- Forward Iteration

$$\mathbf{V} = \mathbf{G}\mathbf{S}\dot{\mathbf{q}} \quad (4.10)$$

$$\dot{\mathbf{V}} = \mathbf{G}_{T_0}\dot{\mathbf{V}}_0 + \mathbf{G}\mathbf{S}\ddot{\mathbf{q}} + \mathbf{G}\mathbf{A}_1\mathbf{V} \quad (4.11)$$

- Backward Iteration:

$$\dot{\mathbf{V}} = \mathbf{G}^T\mathbf{F}^E + \mathbf{G}^T\mathbf{M}\dot{\mathbf{V}}_0 + \mathbf{G}^T\mathbf{A}_2\mathbf{M}\mathbf{V} \quad (4.12)$$

$$\boldsymbol{\tau} = \mathbf{S}^T\mathbf{F} \quad (4.13)$$

- $\mathbf{V}$  and  $\dot{\mathbf{V}}$ :  
Generalized body velocity and acceleration column vector  $\mathbf{V} = [V_1, V_2, \dots, V_n]$  and  $\dot{\mathbf{V}} = [\dot{V}_1, \dot{V}_2, \dots, \dot{V}_n] \in R^{6 \times 6}$ .
- $\dot{\mathbf{V}}_0$ :  
Generalized body velocity  $\dot{\mathbf{V}}_0 = (0, 0, g, 0, 0, 0)^T \in R^{6 \times 1}$  of the base module.
- $\mathbf{G}_{T_0}$ :  
Adjoint Representation  $\mathbf{G}_{T_0} \in R^{6n \times 6}$ .

$$\mathbf{G}_{T_0} = \begin{bmatrix} Ad_{T_{01}^{-1}} \\ Ad_{T_{02}^{-1}} \\ Ad_{T_{03}^{-1}} \\ \vdots \\ Ad_{T_{0n}^{-1}} \end{bmatrix} \in R^{6n \times 6} \quad (4.14)$$



Equation 4.14 shows the general format of the matrix  $\mathbf{G}_{T_0}$ . Every row contains the computation of transpose of the inverted adjoint representation  $Ad_{T_{0i}^{-1}} = Ad_{T_{0i}}^{-1}$ . To obtain the complete Matrix the adjoint representations of the forward transformation  $T_{0i}$  every module  $i = 0, \dots, n$  with respect to the base module "0" are combined according to the Graph's Traversing Order.

**Definition 4.1 (Adjoint Representation Matrix)**

The *Adjoint Representation Matrix* of a directed kinematic graph  $\vec{\mathcal{G}}$  of a modular robot with  $n + 1$  modules (vertices) is an  $6(n + 1) \times 6$  matrix,  $\mathcal{G}_T(\vec{\mathcal{G}}) = [g_{T_{0i}1}]$ ,  $i = 0, \dots, n$ , such that  $g_{T_{0i}1} = Ad_{T_{0i}^{-1}}$ , where  $T_{0i}$  is the Forward Transformation of module  $i$  relative to the base frame.

• **G :**

Transmission Matrix  $\mathbf{G} \in R^{6n \times 6n}$ .

$$\mathbf{G} = \begin{bmatrix} I_{6 \times 6} & 0 & 0 & \cdots & 0 \\ r_{12}Ad_{T_{12}^{-1}} & I_{6 \times 6} & 0 & \cdots & 0 \\ r_{13}Ad_{T_{13}^{-1}} & r_{23}Ad_{T_{23}^{-1}} & I_{6 \times 6} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{1n}Ad_{T_{1n}^{-1}} & r_{2n}Ad_{T_{2n}^{-1}} & r_{3n}Ad_{T_{3n}^{-1}} & \cdots & I_{6 \times 6} \end{bmatrix} \in R^{6n \times 6n} \quad (4.15)$$

The Transmission Matrix  $\mathbf{G}$  is of supreme importance for the succeeding computation of the closed-form equation of Motion. The scheme of  $\mathbf{G}$  represented in equation 4.15 discloses the involvement of the Accessibility matrix  $\mathcal{R}(\vec{\mathcal{G}})$  and its elements  $r_{ij}$ . If  $r_{ij} = 1$ , the Forward Transformation  $T_{ij}$  from frame of module  $i$  to the one of module  $j$  will be calculated. Subsequently the Adjoint Representation  $Ad_{T_{ij}}$  will be inverted and the calculated  $Ad_{T_{ij}^{-1}}$  will be inserted in  $\mathbf{G}$  in the place  $(j, i)$ .

**Definition 4.2 (Transmission Matrix)**

The *Transmission Matrix* of a directed kinematic graph  $\vec{\mathcal{G}}$  of a modular robot with  $n + 1$  modules (vertices) is an  $6(n + 1) \times 6(n + 1)$  matrix,  $\mathcal{G}(\vec{\mathcal{G}}) = [g_{ij}]$ ,  $i, j = 0, \dots, n$ , such that  $g_{ij} = Ad_{T_{ji}^{-1}}$ , if the corresponding element of the Accessibility Matrix  $\mathcal{R}(\vec{\mathcal{G}})$  equals one,  $r_{ji} = 1$ . If  $i = j$ ,  $g_{ij} = I_{6 \times 6}$ ;  $g_{ij} = 0$ , otherwise.

MATLAB-Code:

```
% Computation of G_T0
% T0j and T0jcell are the Forward Kinematics T0j(q)
```

## 4 Implementation

---

```

T0jcell = cell(1,c);
add = 0;
for rowsPM = 1:p
    prod = 1;
    for y = 1:pathjoints(rowsPM)
        Tij0 = Tij0cell{pos(y+add,1),pos(y+add,2)};
        prod = prod*Tij0*twistexp(S{y+add,y+add},q(y+add));
        if(y == max(pathjoints(rowsPM)))
            T0jpath{rowsPM,1}= prod;
        end
        T0jcell{1,y+add} = prod;
    end
    add=add+pathjoints(rowsPM);
end
T0j = [T_0*twistexp(S{1,1},0);cat(1, T0jcell{:})];
G_T0(6*rows-5:6*rows,1:6)= iadR0(T0j(4*rows-3:4*rows,:));
end
G_T0;
G_T0cell = cell(c,1);
for x = 2:c
    G_T0cell{x,1} = iadR0(T0jcell{1,x});
end
% Precondition for Computation of G
% dyadcell{x,y} = Txy(q(y)) of Frame y relative to x
dyadcell = cell(r,r);
for x = 1:r
    for y = 1:r
        if(y >= x)
            dyadcell{x,y} = zeros(4);
        else
            if(TAM(x,y)==1)
                [dist, path, pred] = shortestpath(bg, y, x);
                if (length(path)>2)
                    prod = 1;
                    for h=1:length(path)-1
                        prod=prod*Tij0cell{path(h),path(h+1)}
                            *twistexp(S{path(h+1),path(h+1)},
                                q(path(h+1)));
                    end
                    dyadcell{x,y}=prod;
                else
                    tij=Tij0cell{y,x}*twistexp(S{x,x},q(x));
                    dyadcell{x,y}=tij;
                end
            end
        end
    end
end

```

## 4 Implementation

---

```

        else
            dyadcell{x,y}=zeros(4);
        end
    end
end
end
end
%%% Calculate the Transmission Cell Gcell %%%
Gcell = cell(r,r);
for x = 1:r
    for y = 1:r
        if (x == y)
            Gcell{x,y} = eye(6);
        elseif (y > x)
            Gcell{x,y} = zeros(6);
        else
            if (TAM(x,y) == 1)
                Gcell{x,y} = TAM(x,y)*iadR0(dyadcell{x,y});
            elseif (TAM(x,y) == 0)
                Gcell{x,y} = zeros(6);
            else
                disp('error occured')
            end
        end
    end
end
end
end

```

- **S:**  
Joint Twist Matrix  $\mathbf{S} = \text{diag}[s_1, s_2, \dots, s_n] \in R^{6n \times n}$ . Twists  $s$  in module coordinates.  
MATLAB-Code:

```

% Generate Twist Coordinates for all Modules
% S = eye(r)*twistcoordinates = Rotation about x-Axes
S=cell(r,r);
for x = 1:r
    for y = 1:r
        if (x == y)
            S{x,y} = [0,0,0,1,0,0]';
        else
            S{x,y} = [0,0,0,0,0,0]';
        end
    end
end
end
end

```

- $\dot{\mathbf{q}}$  and  $\ddot{\mathbf{q}}$ :

Joint velocity and acceleration column vector  $\dot{\mathbf{q}} = [\dot{q}_1, \dot{q}_2, \dots, \dot{q}_n]$  and  $\ddot{\mathbf{q}} = [\ddot{q}_1, \ddot{q}_2, \dots, \ddot{q}_n] \in \mathbb{R}^{n \times 1}$ .

- $\mathbf{A}_1$  and  $\dot{\mathbf{A}}_2$  :  
Adjoint representations  $\mathbf{A}_1 = \text{diag} [-ad_{s_1\dot{q}_1}, -ad_{s_2\dot{q}_2}, \dots, -ad_{s_n\dot{q}_n}]$  and  $\mathbf{A}_2 = \text{diag} [-ad_{V_1}^T, -ad_{V_2}^T, \dots, -ad_{V_n}^T] \in \mathbb{R}^{6n \times 6n}$ .
- $\mathbf{M}$ :  
Represents the combined generalized Mass Matrix  $\mathbf{M} = \text{diag} [M_1, M_2, \dots, M_n] \in \mathbb{R}^{6n \times 6n}$ .
- $\mathbf{F}$  and  $\mathbf{F}^E$  :  
The body wrench vector  $\mathbf{F} = [F_1, F_2, \dots, F_n]$  and the external wrench vector  $\mathbf{F}^E = [F_1^e, F_2^e, \dots, F_n^e] \in \mathbb{R}^{6n \times 1}$ .

### Equation of Motion

By combining the equations 4.10, 4.11, 4.12 and 4.13 the closed-form equation of motion for the modular robot can be obtained.

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{N}(\mathbf{q}) = \boldsymbol{\tau} \quad (4.16)$$

- $\mathbf{M}(\mathbf{q})$ : is the Mass Matrix

$$\mathbf{M}(\mathbf{q}) = \mathbf{S}^T \mathbf{G}^T \mathbf{M} \mathbf{G} \mathbf{S} \quad (4.17)$$

- $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ : is the Centrifugal and Coriolis Acceleration

$$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}^T \mathbf{G}^T (\mathbf{M} \mathbf{G} \mathbf{A}_1 + \mathbf{A}_2 \mathbf{M}) \mathbf{G} \mathbf{S} \quad (4.18)$$

- $\mathbf{N}(\mathbf{q})$ : is the Gravitational and External Force

$$\mathbf{N}(\mathbf{q}) = \mathbf{S}^T \mathbf{G}^T \mathbf{M} \mathbf{G}_{T_0} \dot{\mathbf{V}}_0 + \mathbf{S}^T \mathbf{G}^T \mathbf{F}^E \quad (4.19)$$

MATLAB-Code:

```
% Computes Motion Equations with symbolic data
M = S' * G' * GM * G * S;
C = S' * G' * (GM * G * A1 + A2 * GM) * G * S;
Grav = S' * G' * GM * G_T0 * V0dot;
FExt = S' * G' * Fe;
N = Grav + FExt;
invMG = inv(sym(MG));
qddot = invMG * (tau' - (C * qdot' + N));
```

## 4 Implementation

---

As a conclusion the main steps of the generation of dynamical model for any branching type modular robot is illustrated in the next Figure.

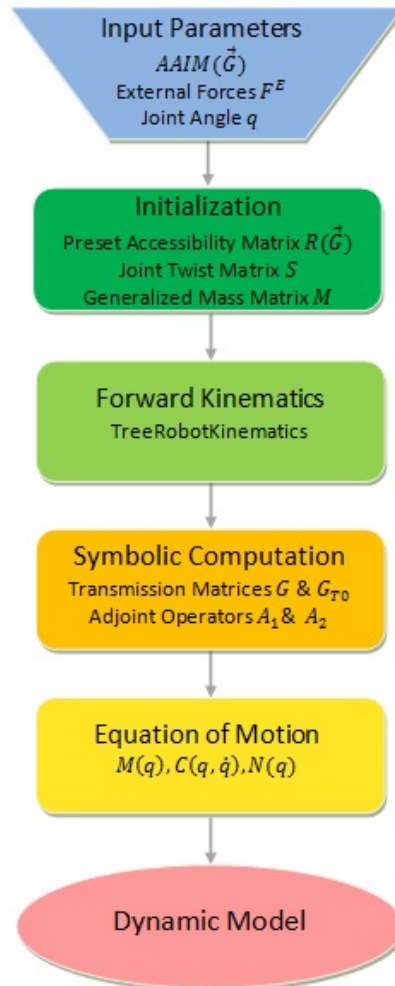


Figure 4.3: DynamicModelGeneration

#### 4.1.4 Comparison of Modular Robot Assembly Prototypes

To demonstrate the capabilities of the framework four typical modular robot assemblies and their geometrical representation forms are listed in the following tables. Additionally all the computation even the results of the Dynamical Model Generation of the elementary modular robot incidence, the dyad, are presented.



Modular Robot Assembly	Directed Graph
	
Adapted Assembly Incidence Matrix	Connectivity Matrix
$AAIM = \begin{matrix} & e_1 & e_2 & Links \\ v_0 & \begin{pmatrix} 3 & 0 \\ 4 & 3 \\ 0 & 4 \end{pmatrix} & \begin{matrix} B \\ C \\ C \end{matrix} \\ Joints & \begin{pmatrix} SP & SP \\ & 0 \end{pmatrix} & \end{matrix}$	$CM = \begin{matrix} & v_0 & v_1 \\ v_0 & \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \\ v_1 & \end{matrix}$
Accessibility Matrix	Path Matrix
$R = \begin{matrix} & v_0 & v_1 \\ v_0 & \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \\ v_1 & \end{matrix}$	$PM = \begin{matrix} & v_0 & v_1 \\ p_1 & \begin{pmatrix} 1 & 1 \end{pmatrix} \end{matrix}$

Table 4.1: Serial Type Modular Robot Assembly with  $n = 3$  Modules.

## 4 Implementation


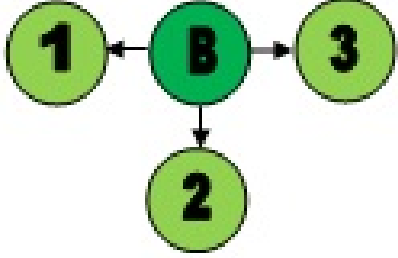
Modular Robot Assembly	Directed Graph
	
Adapted Assembly Incidence Matrix	Connectivity Matrix
$AAIM = \begin{matrix} & e_1 & e_2 & e_3 & L \\ v_0 & \begin{pmatrix} 3 & 2 & 4 & B \end{pmatrix} \\ v_1 & \begin{pmatrix} 4 & 0 & 0 & C \end{pmatrix} \\ v_2 & \begin{pmatrix} 0 & 3 & 0 & C \end{pmatrix} \\ v_3 & \begin{pmatrix} 0 & 0 & 3 & C \end{pmatrix} \\ J & \begin{pmatrix} SP & SO & SP & 0 \end{pmatrix} \end{matrix}$	$CM = \begin{matrix} & v_0 & v_1 & v_2 & v_3 \\ v_0 & \begin{pmatrix} 0 & 1 & 1 & 1 \end{pmatrix} \\ v_1 & \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix} \\ v_2 & \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix} \\ v_3 & \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix} \end{matrix}$
Accessibility Matrix	Path Matrix
$R = \begin{matrix} & v_0 & v_1 & v_2 & v_3 \\ v_0 & \begin{pmatrix} 0 & 1 & 1 & 1 \end{pmatrix} \\ v_1 & \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix} \\ v_2 & \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix} \\ v_3 & \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix} \end{matrix}$	$PM = \begin{matrix} & v_0 & v_1 & v_2 & v_3 \\ p_1 & \begin{pmatrix} 1 & 1 & 0 & 0 \end{pmatrix} \\ p_2 & \begin{pmatrix} 1 & 0 & 1 & 0 \end{pmatrix} \\ p_3 & \begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix} \end{matrix}$

Table 4.2: Branching Type Modular Robot Assembly with  $n = 4$  Modules.

## 4 Implementation


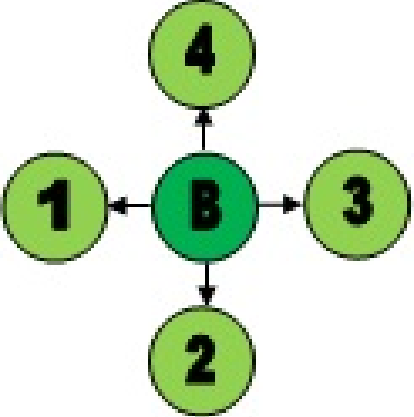
Modular Robot Assembly	Directed Graph
	
Adapted Assembly Incidence Matrix	Connectivity Matrix
$  \mathcal{AAM} = \begin{matrix} & e_1 & e_2 & e_3 & e_4 & L \\ \begin{matrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ J \end{matrix} & \begin{pmatrix} 3 & 2 & 4 & 5 & B \\ 4 & 0 & 0 & 0 & C \\ 0 & 3 & 0 & 0 & C \\ 0 & 0 & 3 & 0 & C \\ 0 & 0 & 0 & 3 & C \\ SP & SO & SP & SO & 0 \end{pmatrix} \end{matrix}  $	$  \mathcal{CM} = \begin{matrix} & v_0 & v_1 & v_2 & v_3 & v_4 \\ \begin{matrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{matrix} & \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{matrix}  $
Accessibility Matrix	Path Matrix
$  \mathcal{R} = \begin{matrix} & v_0 & v_1 & v_2 & v_3 & v_4 \\ \begin{matrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{matrix} & \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{matrix}  $	$  \mathcal{PM} = \begin{matrix} & v_0 & v_1 & v_2 & v_3 & v_4 \\ \begin{matrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{matrix} & \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} \end{matrix}  $

Table 4.3: Branching Type Modular Robot Assembly with  $n = 5$  Modules.



## 4 Implementation

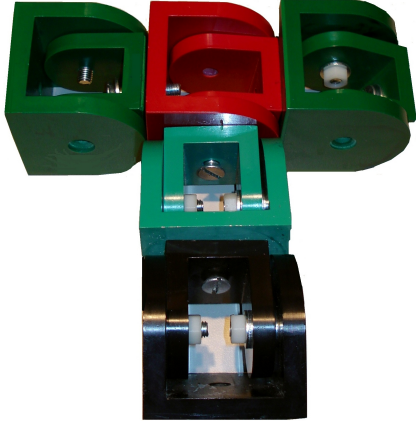
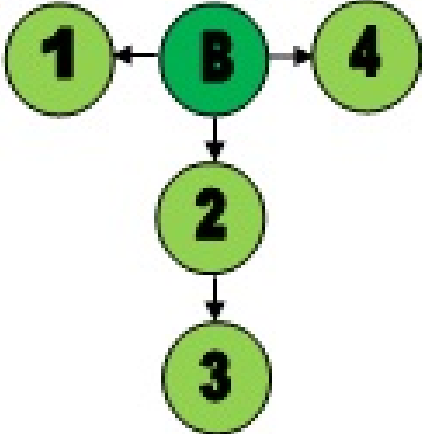
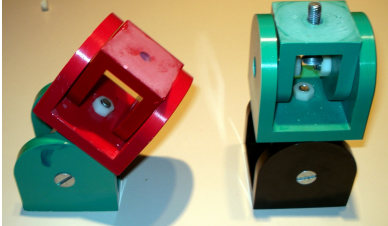
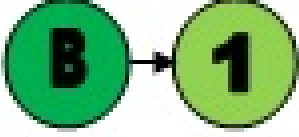
Modular Robot Assembly	Directed Graph
	
Adapted Assembly Incidence Matrix	Connectivity Matrix
$  \mathcal{A} \mathcal{I} \mathcal{M} = \begin{matrix} & e_1 & e_2 & e_3 & e_4 & L \\ \begin{matrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ J \end{matrix} & \begin{pmatrix} 4 & 2 & 0 & 3 & B \\ 4 & 0 & 0 & 0 & C \\ 0 & 3 & 4 & 0 & C \\ 0 & 0 & 3 & 0 & C \\ 0 & 0 & 0 & 4 & C \\ SP & SO & SP & SP & 0 \end{pmatrix} \end{matrix}  $	$  \mathcal{C} \mathcal{M} = \begin{matrix} & v_0 & v_1 & v_2 & v_3 & v_4 \\ \begin{matrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{matrix} & \begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{matrix}  $
Accessibility Matrix	Path Matrix
$  \mathcal{R} = \begin{matrix} & v_0 & v_1 & v_2 & v_3 & v_4 \\ \begin{matrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{matrix} & \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{matrix}  $	$  \mathcal{P} \mathcal{M} = \begin{matrix} & v_0 & v_1 & v_2 & v_3 & v_4 \\ \begin{matrix} p_1 \\ p_2 \\ p_3 \end{matrix} & \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} \end{matrix}  $

Table 4.4: Branching Type Modular Robot Assembly with  $n = 5$  Modules.

## 4 Implementation

Modular Robot Assembly	Directed Graph
	
Adapted Assembly Incidence Matrix	Connectivity Matrix
$AAIM = \begin{matrix} & e_1 & Links \\ v_0 & \begin{pmatrix} 3 & B \\ 4 & C \end{pmatrix} \\ Joints & \begin{pmatrix} SP & 0 \end{pmatrix} \end{matrix}$	$CM = \begin{matrix} & v_0 & v_1 & v_2 \\ v_0 & \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \\ v_1 & & & \\ v_2 & & & \end{matrix}$
Accessibility Matrix	Path Matrix
$\mathcal{R} = \begin{matrix} & v_0 & v_1 & v_2 \\ v_0 & \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \\ v_1 & & & \\ v_2 & & & \end{matrix}$	$\mathcal{PM} = p_1 \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$
Adjointrepresentation	
$\mathbf{G}_{T_0} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 \\ \cos(q_1) & 0 & \sin(q_1) & l \sin(q_1) & 0 & -1l \cos(q_1) & 0 \\ -1 \sin(q_1) & 0 & \cos(q_1) & l \cos(q_1) & 0 & l \sin(q_1) & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & \cos(q_1) & 0 & \sin(q_1) & 0 \\ 0 & 0 & 0 & -\sin(q_1) & 0 & \cos(q_1) & 0 \end{pmatrix}$	
Transmission Matrix	
$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \cos(q_2) & 0 & \sin(q_2) & l \sin(q_2) & 0 & -1l \cos(q_2) & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 \sin(q_2) & 0 & \cos(q_2) & l \cos(q_2) & 0 & l \sin(q_2) & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \cos(q_2) & 0 & \sin(q_2) & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \sin(q_2) & 0 & \cos(q_2) & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	

## 4 Implementation

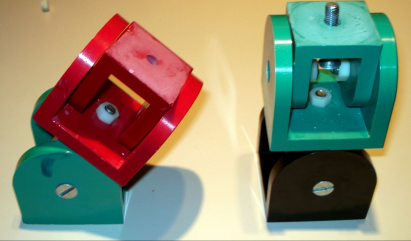
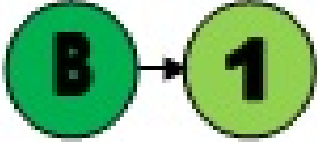
Modular Robot Assembly	Directed Graph
	
Mass Matrix	
$\mathbf{M}(\mathbf{q}) = \begin{pmatrix} \frac{3 \cos(10)^2}{250} + \frac{3 \sin(10)^2}{250} + \frac{1}{500} & 0 \\ 0 & \frac{1}{500} \end{pmatrix}$	
Coriolis and Centrifugal Acceleration	
$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$	
Gravitational and External Forces	
$\mathbf{N}(\mathbf{q}) = \begin{pmatrix} 0.3 \sin(5) - 0.3 \sin(10) + \cos(5) \cos(10) + 0.3 \cos(10) \sin(10) + \sin(5) \sin(10) \\ 0.3 \sin(10) \end{pmatrix}$	
Closed-Form Equation of Motion	
$\boldsymbol{\tau} = \begin{pmatrix} 0.3 \sin(5) - 0.3 \sin(10) + \cos(5) \cos(10) + 0.3 \cos(10) \sin(10) + \sin(5) \sin(10) \\ 0.3 \sin(10) + 10q_2 - 1 \text{ g l m my } \sin(q_2) \end{pmatrix}$	

Table 4.5: Dynamics and Kinematics of a Dyad with  $n = 2$  Modules.

This was an illustrating example of the framework and its ability to generate a model of kinematics and dynamics for any modular robot. The smallest possible configuration was chosen as bigger assemblies' matrices exceed the margin. Still it is a suitable demonstration and representation of the most convenient way to obtain a kinematic and dynamical model for modular robot assemblies.

## 5 Conclusion

This thesis provides a firm basis in automatically developing models for kinematics and dynamics of modular robots assemblies. The elaborated Matlab framework based on screw theory and the adaptations to the theoretical approach shall provide further promising steps towards fast adapting modular robots which are suitable for multiple tasks.

It is desirable that future researchers continue these directions of screw theory based calculations and examine thoroughly inverse kinematics to achieve a foundation to develop an optimal control mechanism, such that in the near future multi modular robots are able to precisely follow desired trajectories.

### 5.1 Summary

In the course of this thesis the framework of an automated model generator for dynamics and kinematics of reconfigurable modular robot systems is elaborated to meet the requirements of applicable and suitable models for platforms that are not yet as idealized as postulated or assumed in articles and theoretical examinations.

Particularly the focus of attention was drawn on the questions: How to implement a correct representation of the platforms "Scout" and "Backbone" regarding their extraordinary characteristic cube modules? And on account of these deviations of shape and functionality, which alternations need to be carried into execution?

These questions were answered by the introduced Adapted Assembly Incidence Matrix, by modified joint representations and furthermore by restrictions and connecting rules.

Another idea was to support the finding of accurate models by supplying an overview of different connection varieties, which was realized by the look-up table.

Above all stands the development of the automatic model generator, which provides a step by step solution for simulation and control purposes.

In contrast to many other comparable multi robot organism approaches, this thesis examines the kinematics and dynamics of modular robots founded on screw theory to face the huge challenge of modeling systems of high complexity, where traditional model generation methods fail or deliver unsatisfying or incomplete results due to the immense number of degrees of freedom.

This framework is able to supply the user with the specified modular robot representation forms by calculating the dyads' Forward Transformations depending on various calculations based on the Assembly Incidence Matrix. Additionally the recursive Newton-Euler

Algorithm is applied to specify the dynamics of the modular robot incidence. Both kinematic and dynamical formulations are assembled to obtain the closed form equations of motions, which is the purpose of this simulation process.

### 5.2 Further research

During the elaboration process of this thesis new thoughts and ideas entered my mind, but the predefined time to work on a Bachelor thesis is very limited and it is not possible to pursue all of them. Additionally to those concepts which were already mentioned in the former chapters of this thesis, the below ideas are presented to arouse deeper interest in the research topic of modular robots:

- **Inverse Kinematics:**  
Tree typed modular robots need a special treatment and strategies to solve all differential kinematic equation simultaneously and to obtain the solution for a given pose, [Chen and Yang, 1999].
- **Random Friction Models**  
One favorable approach is the least square observer which has to be constructed for the online estimation of friction dynamics. A perturbed system model governing the friction estimation error is to be derived and a Model Reference Adaptive Control has to be derived to mitigate its effect. Also, stability conditions for the perturbed system model using the Lyapunov stability theory can be applied, [Cho et al., 2009].
- **Feedback Linearization:**  
Robot Systems are often highly complicated non linear systems. In order to control them, the nonlinearities have to be converted into preferably realistic linear systems without greater loss of information since common control methods involve linear systems.
- **Optimal Position and Trajectory Control**  
As mentioned trajectory and position control is a very interesting topic, which lead to immense improvement of the functionality of modular robots and amplifies the range of possible applications. Control techniques may be robust adaptive or optimal controllers.

Last but not least the framework of this thesis will be supplemented by another thesis, which is effectuation an Graphical User Interface, which allows the user to create Modular Robot Assemblies graphically and supply this framework with the belonging Assembly Incidence Matrix to commence the calculation of simulation parameters.

# Bibliography

- [Aldebaran, 2010] Aldebaran (2010). [http://www.hizook.com/files/users/3/Nao\\_H25\\_Robot\\_Aldebaran.jpg](http://www.hizook.com/files/users/3/Nao_H25_Robot_Aldebaran.jpg). (September 2, 2010).
- [AnnaKonda, 2006] AnnaKonda (2006). [http://www.robotonline.net/upload/gallery/2010/11/anna\\_konda\\_large.jpg](http://www.robotonline.net/upload/gallery/2010/11/anna_konda_large.jpg). (November 17, 2010).
- [Asimo, 2004] Asimo (2004). <http://www.aoyama.ru/img/asimo.jpg>. (December 15, 2004).
- [Ball, 1876] Ball, R. S. (1876). *The theory of screws - a study in the dynamics of a rigid body*. Hodges.
- [Chen and Burdick, 1995] Chen, I.-M. and Burdick, J. (1995). Determining task optimal modular reconfigurable robot assembly configurations. *IEEE International Conference on Robotics and Automation, Nagoya, Japan*.
- [Chen and Yang, 1996] Chen, I.-M. and Yang, G. (1996). Configuration independent kinematics for modular robots. *IEEE Int. Conf. Robotics and Automation, Minneapolis, MN*.
- [Chen and Yang, 1997] Chen, I.-M. and Yang, G. (1997). Automatic model generation for modular reconfigurable robot dynamics. *2nd Asian Control Conference, Seoul, Korea*.
- [Chen and Yang, 1998] Chen, I.-M. and Yang, G. (1998). Automatic model generation for modular reconfigurable robot dynamics. *Journal of Dynamic Systems, Measurement, and Control*, 120.
- [Chen and Yang, 1999] Chen, I.-M. and Yang, G. (1999). Numerical inverse kinematics for modular reconfigurable robots. *Journal of Robotics Systems*, 16.
- [Cho et al., 2009] Cho, H., Fadali, M., Lee, K., and Kim, N. (2009). Adaptive position and trajectory control of autonomous mobile robot systems with random friction. *IET Control Theory and Applications*.
- [CKBot, 2009] CKBot (2009). <http://www.ros.org/news/2010/04/robots-using-ros-modlabs-ckbots.html>. (April 5, 2010).
- [Davidson et al., 2004] Davidson, J. K., Hunt, K. H., and Davidson, J. K. (2004). *Robots and screw theory - applications of kinematics and statics to robotics*. Oxford University Press, New York.
- [Deo, 1974] Deo, N. (1974). *Graph theory with applications to engineering and computer science*. Prentice-Hall.

- [iMobot, 2011] iMobot (2011). <http://spectrum.ieee.org/image/1820972.jpg>. (March 28, 2011).
- [Kernbach et al., 2011] Kernbach, S., Schlachter, F., Humza, R., Liedke, J., Popesku, S., Russo, S., Ranzani, T., Manfredi, L., Stefanini, C., Matthias, R., Schwarzer, C., Girault, B., Alschbach, P., and E. Meister, O. (2011). Heterogeneity for increasing performance and reliability of self-reconfigurable multi-robot organisms. *IROS11, workshop on "Reconfigurable Modular Robotics"*, San Francisco, 2011.
- [Meister, 2007] Meister, E. (2007). Development of CAN-based Computational Cells for Multi Robot Organisms. *Diplomarbeit*, 1025.
- [Multi-robot, 2008] Multi-robot (2008). <http://www.symbrion.org/>. (August 29, 2008).
- [Murray and Li, 1994] Murray, R. M. and Li, Z. (1994). *A mathematical introduction to robotic manipulation*. CRC Press, 0002. aufl. edition.
- [Qrio, 2006] Qrio (2006). [http://www.reckmann.org/wp-content\\_qrio\\_aibo.jpg](http://www.reckmann.org/wp-content_qrio_aibo.jpg). (January 26, 2006).
- [Rothermel, 2008] Rothermel, M. (2008). Mechanical design and assembly of a swarm robot with docking capabilities for building multi-robot organisms. *Studienarbeit*.
- [Siciliano and Khatib, 2008] Siciliano, B. H. and Khatib, O. H. (2008). *Springer handbook of robotics*. Springer, Wiesbaden, 1. aufl. edition.
- [Sohl and Bobrow, 2001] Sohl, G. A. and Bobrow, J. E. (2001). A recursive multibody dynamics and sensitivity algorithm for branched kinematic chains. *Journal of Dynamic Systems, Measurement, and Control*, 123.
- [SYMBRION, 2008] SYMBRION (2008). <http://www.symbrion.eu>. (August 29, 2008).
- [Wikipedia, ] Wikipedia.
- [Wikipedia, 2008] Wikipedia (2008). <http://www.wikipedia.org>. (August 29, 2008).
- [Wurst, 1986] Wurst, K. H. (1986). *The Conception and Construction of a Modular Robot System*. Proc. 16th Int. Sym. Industrial Robotics.
- [Yu and Chen, 2002] Yu, Q. and Chen, I.-M. (2002). A general approach to the dynamics of nonholonomic mobile manipulator systems. *Journal of Dynamic Systems, Measurement, and Control*, 124.

## **Declaration**

All the work contained within this thesis, except where otherwise acknowledged, was solely the effort of the author. At no stage was any collaboration entered into with any other party.

---

(Jutta Ganzhorn)