Kinematics of 3D Folding Structures for Nanostructured Origami™

Diplomarbeit

submitted by

cand.-ing. Tilman Buchner

December, 2003

Massachusetts Institute of Technology

Department of Mechanical Engineering

3D Optical Systems Group

Prof. George Barbastathis, PhD

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

Werkzeugmaschinenlabor

Steuerungstechnik und Automatisierung

Prof. Dr.-Ing. Dr.-Ing. E.h. M. Weck

Advisor: Dipl.-Ing. Frank Possel-Dölken

Abstract

The 3D Optical Systems Group at MIT investigates Nanostructured Origami™ 3D fabrication and assembly. The idea is to assemble complex hybrid (chemical or biological reactors, optical sensing, digital electronic logic, mechanical motion) systems in 3D by using exclusively 2D lithography technology. The 3D shape is obtained by folding the initial 2D membrane in a prescribed way, in a manner reminiscent of the Japanese art of origami (paper-folding). The patterning method (2D nanolithography, nanoimprinting and other techniques) as well as the actuation principle (Lorentz force actuation) which is responsible for initializing the folding process have already been developed and established.

The knowledge of the dynamic folding process itself, needed to reach any desired 3D shape from a 2D initial state is to-date unexplored. Hence, the primary objective of this thesis is to determine the motions required to reach the goal (folded state) from a given initial state (unfolded). Hereto general folding operations will be analyzed and a new method to describe its kinematics for any arbitrary structure will be developed.

We propose to use origami mathematics in combination with kinematical formulations in the area of robotics and gearing. The most attractive feature of origami is that one can construct a wide variety of complex shapes using a few axioms, simple fixed initial conditions and one mechanical operation, a fold. In a second step, the idea will be pursued to transfer this paper folding concept to a more generally admitted approach which uncouples the paper aspect from the folding operation. Hereby a combination of bodies, which are connected by joints, is used to describe the crease structure.

As a result, the crease structure is represented in terms of a closed-loop Multi Body System (MBS) with the property that a change of relative motion at one location induces a change of relative motion elsewhere. To describe this system, a well-known mathematical method, called screw calculus will be applied. Screw calculus is based on Poinsot's result that an arbitrary rigid body motion can be described in terms of a translation cascaded with a rotation around the translation axis. The "screw" then is a matrix exponential describing the motion. By cascading several screws together one can describe the motion of articulated rigid body systems, such as robotic manipulators and, in our case, origami.

The results of the theoretical investigation will be implemented in a design and simulation software tool. The application has features to verify and create folding structures as well as to visualize the folding operation on the basis of the above mentioned kinematics approach.

List of Figures

Table of Contents

1 Introduction

The 3D Optical Systems Group at MIT investigates a new manufacturing method to assemble three-dimensional (3D) intelligent microsystems with complex hybrid functions like chemical or biological reactors, optical sensing, digital electronic logic, mechanical motion for movement or energy harvesting, with nanoscale components from lower dimensional (2D) elements.

The development of current architectures for VLSI (Very Large Scale Integration) integrated circuits and microelectromechanical systems (MEMS) are quickly reaching a point where 2D structures have become utilized and can no longer become denser. In the realm of microelectronics, the 3rd dimension promises a large spectrum of opportunities in numerous technological domains when feature size of planar electronics reaches its minimum physical limit [Jurg 03].

The interactions between the different modalities and its small form factor with a nanoscale of less then 100nm predestine this technology to new physical properties and hence unprecedented functional capabilities. Three dimensional MEMS devices will see wide use in both military and commercial areas, with applications ranging from automobiles and fighter aircraft to printers and munitions [Tang 00]. The Defense Advanced Research Projects Agency (DARPA), for example, is soliciting research proposals in the area of MEMS with the object to co-locate the functions of sense, computation, actuation, control, communication, and power. From the integration of these functions into macro systems they expect a radical change concerning the way people and machines interact with the physical world [Micro 98].

As an example of an intelligent microsystem that could be fabricated with this new manufacturing method, a biochemical sensor can be considered. This would require the following components [Barb 03]:

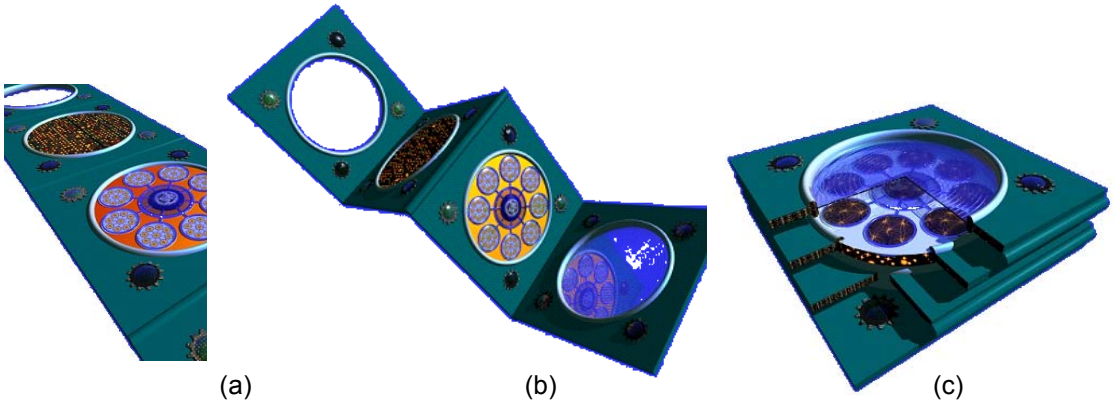
- the top of the structure would function as a suction pump which captures particle specimens from the environment;
- the next recipient of the captured particles would be a biochemical reactor which would produce a clearly identifiable chemical result depending on the nature of the captured particles
- optical sensing (a miniature high-resolution spectrometer) would be used to detect the results of the bio-chemical reaction
- digital electronics would process the results of the detection, and

- an antenna would establish a wireless connection via which the results would be communicated to the users.

In addition the system might be self-powered by harvesting vibrational energy from disturbances, and it might be reconfigurable e.g. changing the pump “nozzle” diameter mechanically, depending on the desirable average size of particles to be tested.

To clarify the challenge to manufacture and assemble such highly complex microsystems we can consult the analogy to the familiar macro world by considering the construction and assembly process of a common car engine. The car engine contains diverse hybrid functions e.g. pumps, electrical chargers, combustion chambers, transducers to mechanical energy, cooling systems, sensors and controls. Industrial robots are responsible for the assembly and interconnectivity of these different components in the correct order and position. The important difference between this kind of assembly and assembly of systems at the micro-nano scale is that tools for conventional manipulations are not able to build nano components with both sufficient accuracy and throughput even for an acceptable price. Hence we keep track of a newly developed technique which is reminiscent of the Japanese art of “origami”, or paper folding that can be thought of as a “templated self-assembly” process called Nanostructured Origami™ 3D fabrication and assembly process, see Figure 1-1.

The idea of self-assembly is to decouple the fabrication process into two parts. During the first fabrication step, features (hybrid functions) as well as creases, which act as joints, are defined on a large membrane-like surface, whereby exclusively standard 2D lithography tools are used.



(a) (b) (c)

Figure 1-1: Nanostructured Origami™ 3D Fabrication and Assembly Process: (a) 1st step, nanopatterning the membrane surfaces; (b) 2nd step, folding the surface along creases and using actuators defined during the 1st step; (c) final folded structure.

During the second manufacturing step, the Lorentz force actuation principle is used to initiate the folding process along the creases, which results in the 2D membrane finding itself in its intended final folded structure in 3D space. This method achieves high throughput for highly complex, non-periodic 3D structures and provides the manufacturing process with accuracy, reliability and throughput comparable to those of conventional 2D lithography.

The manufacturing process itself as well as the following actuating principle is already accurately described. Against these results, the knowledge of the dynamic folding process itself, needed to reach any desired 3D shape from a 2D initial state is to-date unexplored. Hence, the primary objective of this thesis is to determine the motions required to reach the goal from a given initial state in order to manufacture more complex 3D structures and to make the Nanostructured Origami™ approach applicable to the industry. Hereto general folding operations will be analyzed and a new method to describe its kinematics for any arbitrary structure will be developed. The results of this design will be implemented in a design and simulation software tool to create, verify and visualize folding structures for Nanostructured Origami™ devices.

1.1 Inspiration for the 3rd Dimension

In everyday life there are numerous examples which take advantage of the 3rd dimension. A view on the two pictures below shows the landscape of Los Angeles and New York City. Noticeable in this context is their diverse method of construction.

Los Angeles is built into the plane by spreading out large areas. A citizen of this city notices this fact by covering long distances to go from point A to point B. New York City on the other hand is characterized by its world famous skyline which is built up into the 3rd dimension.

Los Angeles - 2D landscape

New York City - 3D landscape

Figure 1-2: Diverse methods of construction are reflected in 2D/3D landscapes

To go from point A to point B you can move in horizontal as well as in vertical directions, this fact reduces the time to reach the final goal position significantly, see Figure 1-2. There are a few other examples even in biology which show that every time the proposition of space is low, nature tries to aim for filling 3D space.

At this juncture, one finds an interesting observation that functionally complex 3D compositions are often built up from lower dimensional structures. For example, DNA and protein are 3D structures assembled from 1D strands [AmSo 00]. The analysis of a bark shows that trees grow outward as laminar sheets only one cell thick [Pal 96], see Figure 1-3.

But one of the most powerful examples in nature is the visual cortex. It is part of the cerebral cortex (responsible for higher brain functions) that is responsible for processing visual stimuli and is located at the back of the human brain in the occipital lobe [COR 03]. It is highly specialized for processing information about static and moving objects and is excellent in pattern recognition. In a sense it is a 3D processor about a size of a sugar cube. There are several major sections within the visual cortex, each of which is composed of folded membranes. Dissecting one region one finds that structurally it is a 2D membrane that is folded over in itself many times. Unfolded, it is a sheet measuring approximately four square inches containing more than 150 million neurons which is comparable to a Pentium 4 processor with a billion transistors within 2.2 cm^2. Neurons (i.e. circuitry) are distributed primarily along the length of the membrane and within the 2D topology.

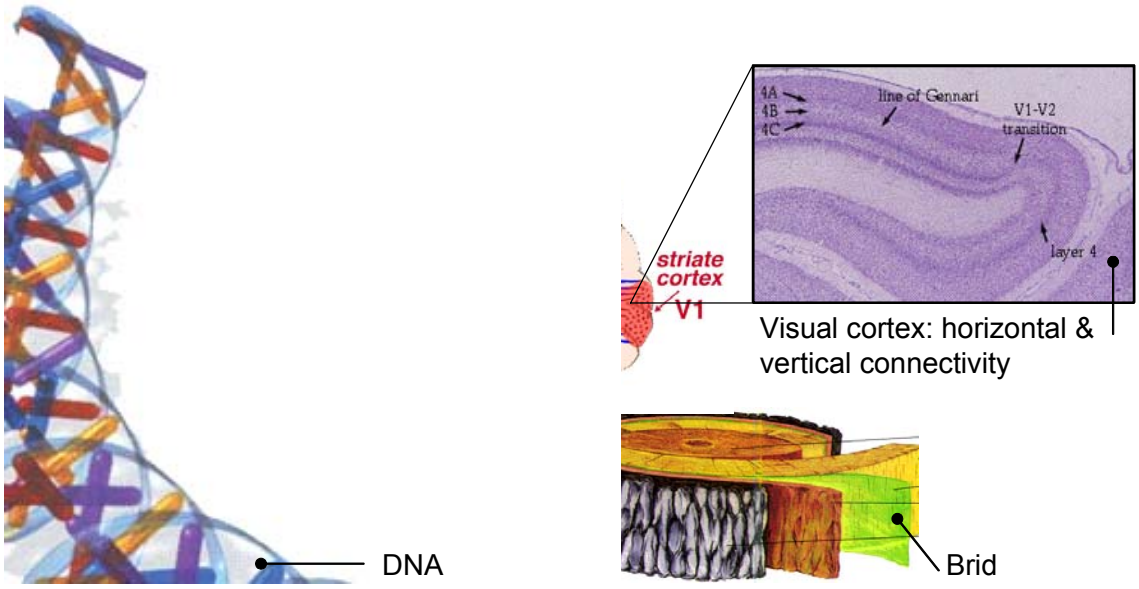


Figure 1-3: Volume filling strategies in biology [Schm 02], [Pal 96]

But neurons also grow between the folded regions of the membrane (vertical), but to a much lesser degree. The final result is an anisotropic, 3D distribution of interconnections [Jurg 03]. One can make the inference that evolution selected toward an anisotropic arrangement of connectivity rather than full isotropic distribution for complexity management reasons [Schm 02]. For the reasons of connectivity and maximization of surface area, the membrane folding concept is attractive for 3D micro- and nanomanufacturing.

1.2 Motivation for Nanostructured Origami™

The evolution of living conditions may serve as an important lesson for the future of 3D micro- and nanosystems. As demonstrated in the illustration of the landscape of Los Angeles and New York City, every time land price is expensive or the population density is enormous, living space was to be built up upwards (buildings are taller than they are wide).

This progression might be the case for MEMS sensors and nano devices over the next decade. The number of active elements squeezed into a planer chip is ever increasing and the demand for smaller consumer electronics will probably never be satiated [Reed 03]. These competing demands necessitate the building of 3D devices just as skyscrapers have evolved and now dominate the urban landscape.

But before we try to transfer the principle of membrane folding to nanotechnology we will examine the two main criteria of microchip fabrication, the manufacturing expenses as well as the problem of heat dissipation which is directly linked with the increase of chip capability. Only if both criteria are satisfied the Nanostructured Origami approach represents a suitable technology for future MEMS fabrication.

1.2.1 Considering the Manufacturing Expenses

In order to induce a change towards new technology there must be a sufficiently important reason to gain market acceptance. Beside new functionality which is the primary motivation, lower costs are followed, in terms of capital investment, by development, and production. Radical changes to machinery or processes are also a severe hurdle because it is very difficult to change the momentum of industry.

A state-of-the art fabrication plant for silicon microchips, for example, now costs approximately $3 billion to be built. On the other hand conventional chip fabrication technology is on a collision course with economics. Today’s best computer chips have silicon features as small as 90 nanometers. But the smaller the features, the

more expensive the optical equipment needed to manufacture those [Tris 03]. Complete mask sets cost millions of dollars. In a layer by layer process, each level requires its own set of unique masks. This means, the cost of the complete mask set will scale directly proportional to the number of layers. At long sight, the microchip industry is forced to develop new technologies.

The membrane folding concept lends oneself to solve this problem. It only requires a single mask set no matter how many layers are to be built. To create these layers it also seeks to use existing microfabrication tools. Additionally, the decoupled process of fabrication and 3D assembly would reduce the development costs and new advances in micro and nanotechnology could be integrated with ease. Considering production, the membrane folding approach represents a time and money saving approach.

1.2.2 Considering the Heat Dissipation

Beside the aspect of costs, one has to take into consideration the importance of heat dissipation for 3D devices. As devices expand into 3rd dimension and increase in volume, the relative amount of surface area decreases. Removing heat becomes more difficult. Especially the development of new generations of wires continues to have smaller cross sectional areas, and more wires are packed closer together. The result is that Joule heating will present some significant challenges to chip designers [Jurg 03]. In the style of nature, the human brain is cooled by circulating fluids. A comparison between our brain which is much more massive than a microprocessor shows that the architecture of our human brain generates only 25 watts, whereas the Pentium 4 in its 2.2 square centimeter package dissipates 80 watts [Lee 02].

This benefit of such an intelligent well dimensioned architecture can be transferred to the nanotechnology by accommodating spacing between layers inside the folded structure. Forced convection for example by blowing air or pumping dielectric fluid through the spacing would constitute an ideal solution for the anticipated heat dissipation problem.

Compendious, the analysis of the most critical points by introducing a new technology are fulfilled and the concept of membrane folding represents a very qualified method which should be pursued in 3D micro and nanotechnology.

The benefits of this approach for building in the 3rd dimension are in a nutshell:

- higher density, volumetric packing of nano features
- Greater connectivity (shorter distance, smaller latency)
- Quick realization of 3D space filling, alignment
- System integration (hybrid components)
- Better thermal management and low power consumption
- Low development and production costs
- Organized architecture
- Seamless integration of new advanced technologies *(*Nanoimprinting, X-Ray and e-Beam lithography)

To point out the value of such nanostructured devices, we refer at this point to the conclusion which discusses future applications of nanostructured MEMS.
In the following chapter the necessary 2D manufacturing technology as well as the connected actuation principle to fabricate wholly integrated 3D systems of sensors and actuators, will be presented. In addition, the conceptional formulation follows in terms of use case analysis for Nanostructured Origami™ in order to determine the functional specification of this thesis.

2 Design of Folding 3D Micro and Nanostructures

In the following we will compare the state-of-the art 3D nanomanufacturing methods and pointing out the advantages of Nanostructured Origami™ as well as the functional requirements which need to be achieved to ensure the product acceptance. Further on the manufacturing process will be described in detail and the actuation principle will be presented. Afterwards the practicability of Nanostructured Origami™ will be explored in experiment results.

2.1 State of the art 3D Nanomanufacturing Methods

The major methods proposed for creating 3D microstructures can be classified as layered vs. self assembly technologies respectively serial vs. parallel methods. The most established methods are the microstereolithography, the two-photon lithiography, the interference lithiography, and self assembling techniques.

The microstereolithography (μSL) technology creates complex 3D shapes in a liquid polymer one layer at a time [ZhJS 99]. A liquid polymer is spread to a thickness ranging from one to ten microns; hereupon a UV laser scans the surface and solidifies the exposed portions. A new, thin layer of polymer is spread and the process is repeated. The benefits are aspect ratios up to 16:1, the ability to create overhung features (resembling cliffs), and the capability to incorporate “exotic” materials such as shape memory alloys and metal powders for use in MEMS actuators [Sent 01].

The limitations are that the UV beam spot size is only 1-2µm, and the vertical slices are limited to the thickness of polymer spread. Crucial is the fact that in this form of manufacturing, fabrication of wires and layering different materials consecutively is not possible, see Figure 2-1.

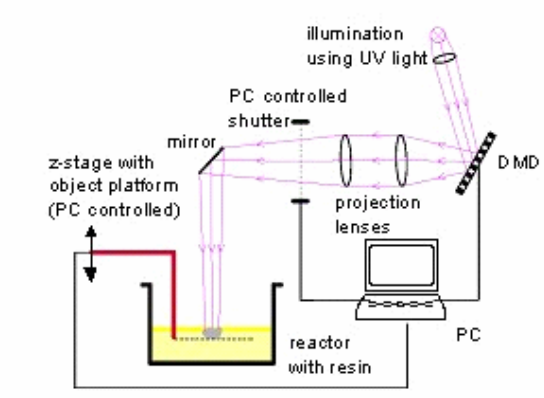


Figure 2-1: The principle of microstereolithography [Jurg 03]

The two-photon lithography is based on a different physical and functional method to create structures in 3D. The polymerization is stimulated by two-photon absorption. Instead of being layer by layer, two-photon writes features with a pulsed laser, a high numerical lens, and linear stages to scan the highly focused spot in three spaces. The properties of this method might someday be useful but the set of material is limited, and creating features a single point at a time in three dimensions can be enormously slow and does not scale well.

A typical example for layer by layer fabrication is Matrix Semiconductor, a pioneer on 3D memory and 3D transistors. Matrix builds its 3D memory very similar to traditional 2D chips, but by stacking memory arrays vertically [Matr 03]. The disadvantage of this technique is that heat becomes a concern as the number of layers increase. Standard 2D chips already require cooling fins and fans to dissipate an average of 80 watts over a 2.2 square centimeters Pentium 5 [Lee 02]. Another approach is the very recent presentation of 3D photo crystals created by assembling individual plates of 2D photonic crystals [AoId+ 03]. The plates are assembled with the aid of a custom micromanipulation system inside of a scanning electron microscope (SEM). Four two twenty plates are brought together and stacked with alignment tolerances of less than 50nm, see Figure 2-2. This method seems very promising, especially considering the alignment reported but the assembly is extremely time-consuming since each plate is handled individually, and manually assembled.

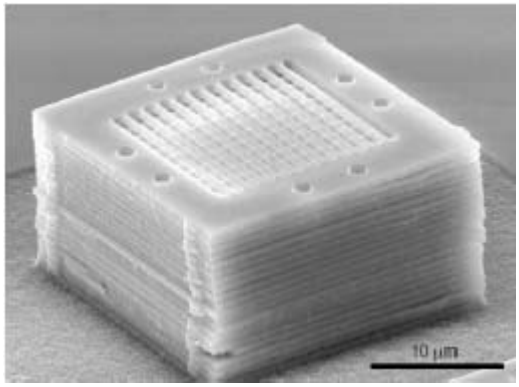


Figure 2-2: 3D photonic crystal fabricated from 2D plates [AoId+ 03]

The Nanostructured Origami^{TM} method which will be described in the following has all the benefits and none of the limitations we have pointed out before. The fundamental advantage is its independence between the 2D fabrication process and the 3D assembly. This permits to utilize all of the existing technologies as well as to incorporate any new advances.

2.1.1 Functional Requirements

To realize the idea of 2D manufacturing and 3D assembly there are five main functional requirements that must be satisfied. The functional requirements are: hinge support, actuation, connectivity, alignment, and latching.

To analyze a useful hinge technology the actuation principle has to be taken into consideration because there is an intended relationship between the hinge and the actuation method. In general you distinguish between chemical, stress based and magnetic actuation principles. To choose a suitable method one needs to keep in mind that the actuation must be compatible with and capable of folding the selected hinge. A few approaches like miniaturizing macroscopic hinges are out of question because of their limitations in the actuation principle. Typically they are using a probe tip to fold the structure, see Figure 2-3. Photoresist joints have also been proposed for rotating silicon micro flaps. Squares of photoresist are patterned between independent segments of silicon. Heating the photoresist, the flaps rotate out of the plane because the surface tension of the photoresist clings to the flaps. This surface tension method represents a coupled design where the hinge and actuation are inseparable. From the axiomatic design point of view this is a suboptimal design because, thermal actuation is highly parallel, to address an individual device is nearly impossible. Other methods like Bilayers, flexible circuits (as common place in laptop computers) or plastically deformed metals are practicable as basic material for hinges.

Other actuating methods like the Permalloy actuation do not support an on or off switching that means that all devices on a wafer are actuated simultaneously. This principle is based on magnetic material patterned on the hinges flaps so that when an external magnetic field is applied, the Permalloy experiences a torque up until the shape anisotropy is aligned with the field, see Figure 2-4.

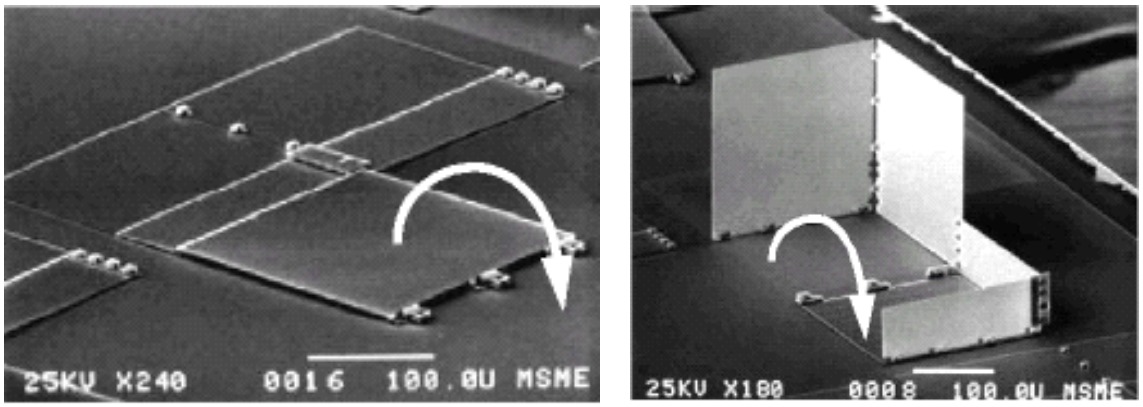


Figure 2-3: Miniaturize macroscopic hinges [Jurg 03]

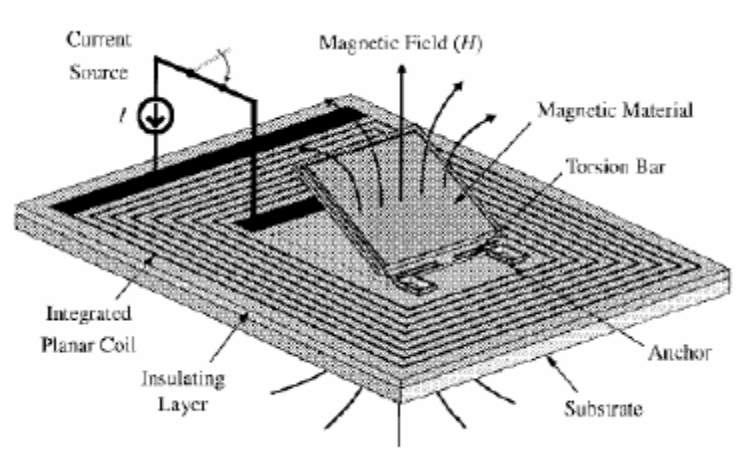


Figure 2-4: Permalloy actuation by integrated coils [Jurg 03]

But the most promising approach is a combination of using plastically deformed metals as hinges which are actuated by a Lorentz force. The Lorentz force has the magnitude advantage that each joint can be actuated individually (highly controllable, addressable, directional (reversing current direction changes the sign of the force), and highly compact because it only requires a wire, see Figure 2-5. The general function of this principle will be explored much more in detail in the following section.

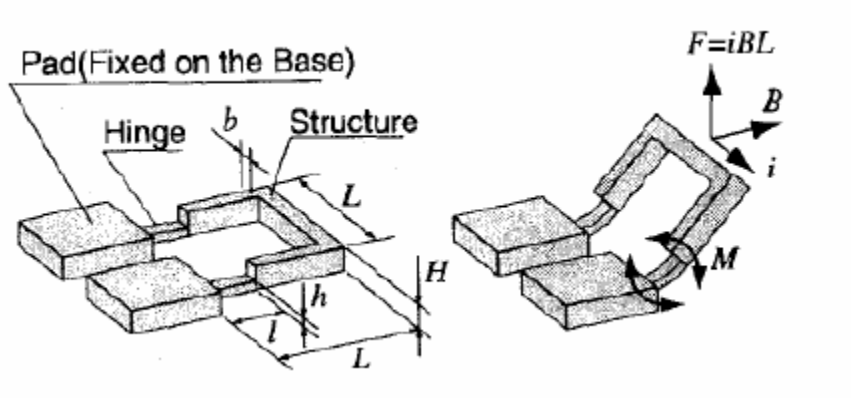


Figure 2-5: University of Tokyo's Lorentz force actuator [IsOH 98]

One of the key features of membrane folding is the ability to run electrical connections even through vertical connections, see Figure 2-6. To adopt this development could be very useful for clock signals or other time-critical signals to reduce significantly the time span which a signal needs to cover a distance from A to B. But connectivity in diverse directions is not limited to electrical signals. Optics presents great opportunities for increased functionality and performance.

There are developments towards using optics to distribute signals throughout a chip using waveguides and free space optics [Jurg 03]. To sum it up, folding opens new possibilities of new chip designs.

If the folding process is completed and the 2D shape has found its final 3D position the alignment of this structure becomes important. A proper placement of the folded membranes with respect to each another is very important if the positioning of optical elements within specified tolerances is crucial in order to guarantee full functionality. An exactly constrained design is necessary ideally that there is no special latching needed. The idea is to generate such a folding structure that alignment tolerances in the order of 10µm are practicable and the structure does not require any manual latching because of its persistence in a self latching position.

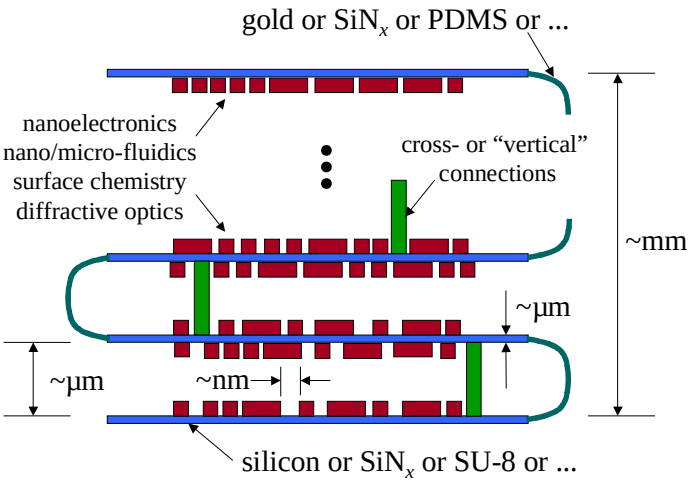


Figure 2-6: Electrical and optical interconnections between layers [BJH+ 03]

These studies demonstrated that in principle it would be possible to fold many wires encased in a flexible polymer in order to transform long 2D chip into a dense 3D structure analogical to the visual cortex model we have mentioned before. Herewith, the five main functions are satisfied and the Nanostructured Origami^{TM} approach lends oneself to represent a suitable manufacturing process for future 3D MEMS applications. In the following the manufacturing process as well as the Lorentz Force actuation method will be described in detail and its applicability is demonstrated through experimental results.

2.2 Fabrication Process

The Nanostructured Origami^{TM} 3D fabrication and assembly process involves two steps; in the first step all devices are fabricated on a planar substrate. Compliant zones are acting as hinges between stiffer regions containing the micro and nano devices. The patterning step and the actuation sequence (i.e. the order in which the folding segments are actuated) must be designed such that every nanofeature on the original 2D membrane finds itself in its intended position in 3D later on.

The batch fabrication inherent in photolithography-based processing makes it possible to fabricate thousands or millions of these components and their “hinges” as easily and within the same time span that it takes to fabricate one component.

The detailed process flow to create such planar substrates comprises three steps and is discussed in the following as well as illustrated in Figure 2-7. Afterwards the actuation principle will be presented and its practicability is shown in terms of an experimental result.

2.2.1 Microfabrication

The primary steps involve KOH (Potassium Oxide Hydrogen) etching through the silicon device (500µm×500µm) layer in order to define the membrane flaps. The silicon segments were patterned by a short, 10µm deep KOH etch to expose the buried oxide. The second step enfolded to deposit and patterns the Au hinges and wires.

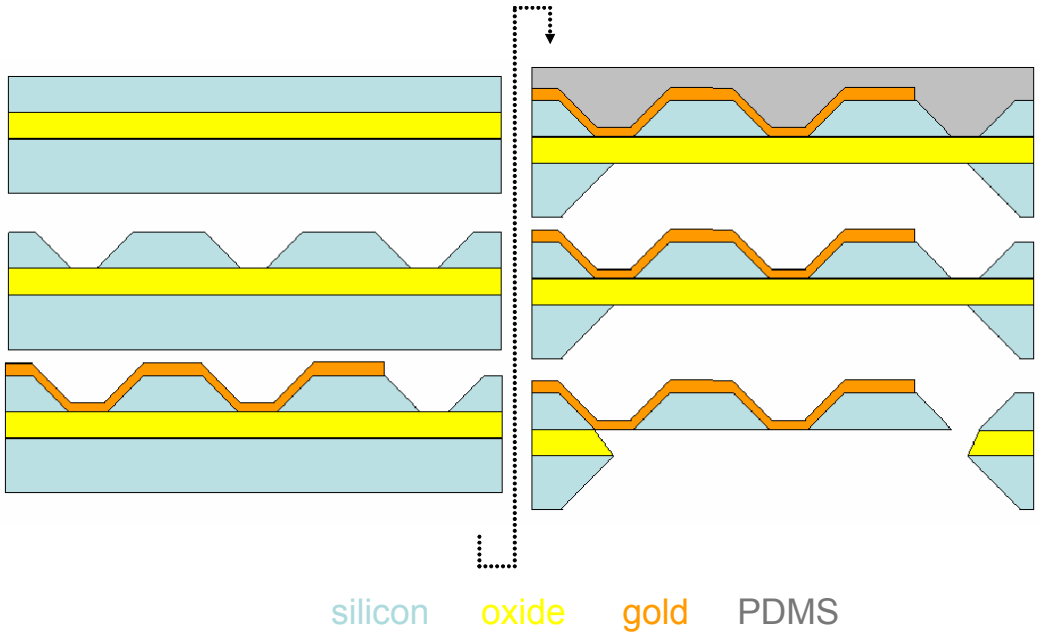


Figure 2-7: Microfabrication process flow

The sloped sidewalls created by the KOH etch were favorable for the Au deposition step, because the smooth surface height variation allowed for a continuous film to be deposited without the need for planarization. The 0.6µm thick Au wires were patterned in a lift-off process using image-reversal photoresist and 30nm of Cr as an adhesion layer. The next major step is the KOH backside etch to avoid possible problems with stiction. During the long KOH backside etch, the devices on the front of the wafer were protected by PDMS. The segment was dry-released using RIE chemistry to selectively etch the buried oxide. All masks were created by means of high resolution (5080 dpi) transparencies [BJH+ 03].
In connection with the microfabrication the second manufacturing step takes place. The actuators are used to initiate a folding procedure, which results in the 2D structure taking its final 3D shape.

2.2.2 Actuation Principle

The actuation principle is based on the Lorentz Force actuation principle. The idea is to place the flat device into a magnetic field B. Hereupon; a current which is oriented perpendicular to the externally applied magnetic field produces a force. This force results in a torque that folds the released segments half-way. The reorienting of the magnetic field results in a complete folding, as shown in Figure 2-8. By reversing the current (or magnetic field), the device can be unfolded if it is desired.
To realize this concept we will use gold (Au) wires as hinges. Hereto, gold hinges serve a dual purpose as actuators using the Lorentz force method as well as interconnections between segments. The advantage of gold as material is that it is plastically deformable, which means the device remains folded after the current is turned off. This in turn contributes greatly to the desired self latching property. Further on it is featured with superior thermal and mechanical properties such as a low Young's modulus $E=80\text{GPa}$, yield strength $\sigma_y=206\text{MPa}$, and a low electrical resistivity $r=2.35\times10^{-8}\ \Omega\text{m}$. Completing, its simple fabrication commends gold as a suitable material for hinges [Jurg 03].
The required bending moment is calculated on the assumption that Gold behaves like an elastoplastic material. This means, the required bending moment increases until the yield occurs. Beyond yield, the moment required for continued bending remains constant.

This interrelation can be shown as:

$$M_{req} = \frac{1}{4} w_h t_h^2 \sigma_y \quad (1.1)$$

Further on, the Lorentz Force is given by $F=w_p i \times B$ (see Right-Hand-Rule). From these equations and the geometry of Figure 2-8, we can deduce that the applied moment is:

$$M_{app} = I_{\max} wLB \cos\Theta \quad (1.2)$$

Whereby Φ is the folding angle. Once the device is fabricated, the applied moment is primarily a function of the current sent trough the wire. The optimal hinge thickness can be found by solving $\max\{M_{app} - M_{req}\}$. A realistic value which is confirmed in numerous experiments is 0,6µm [Jurg 03].

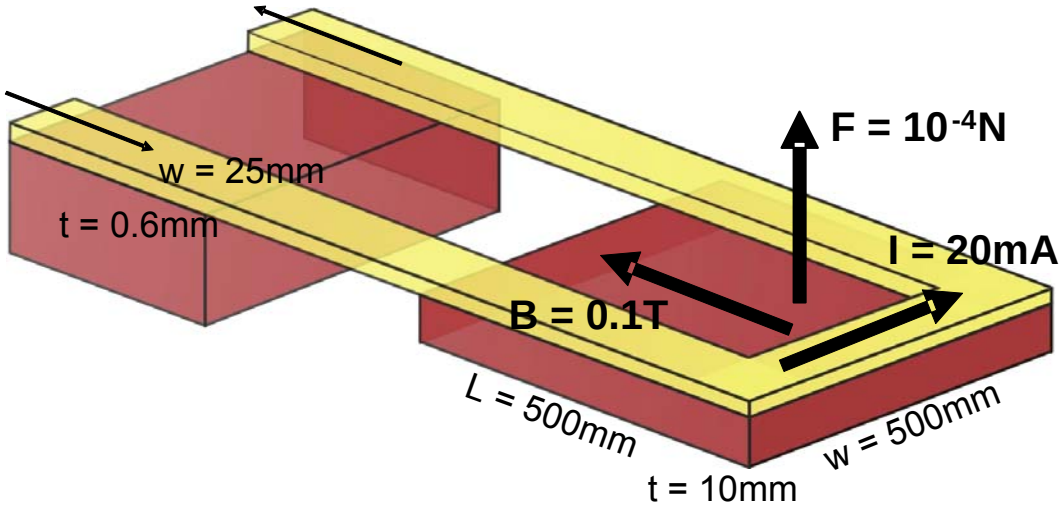


Figure 2-8: Lorentz Force Actuation principle, plastic deformation of hinges

2.3 Experimental Results

Taking these two key technologies (etching, Lorentz Force) as a basis, a large number of experiments have been carried out [Jurg 03]. The devices were fabricated in macroscopic scale with a segment dimension of 500µm x 500µm and 10µm thickness. The folding step takes place within a few milliseconds. We have confirmed in numerous experiments folding to 180 degrees repeatedly with no mechanical or electrical failure actuated by less than 20mA current in a 0.1T magnetic field and with accuracy as good as 10µm.

Figure 2-9 shows a flap folding with respect to a substrate. The flap is blurred because of defocus but it is obvious that a fold by 30°, 90° and 180° is executed successfully. The viewer can extract, e.g., from Figure 2-9(c) a horizontal line which indicates the correctness of the rotation by 90°.

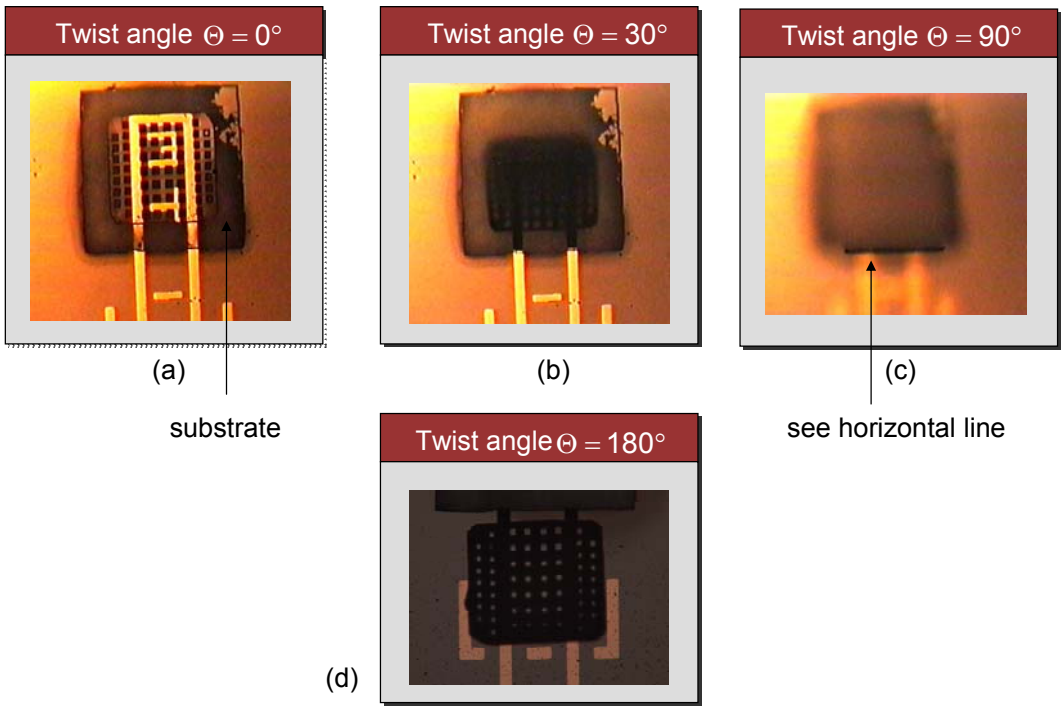


Figure 2-9: Experimental results (magnetic actuation)

3 Conceptual Formulation and Problem Solving

These experimental results reflect the potential of the Nanostructured Origami™ 3D fabrication and assembly process and its great value as a flexible platform for seamless integration of many types of micro and nano fabrication into dense volume. The folding process does not exclude material selection or exotic fabrication requirements. NOFA surpasses all other approaches in terms of 3D connectivity and heat dissipation. Development costs are reduced by fewer masks and parallel fabrication of multiple layers. Capital investment remains in steady state as no new tools are needed because it does not impose any limits on new fabrication technologies. But the essence of creating functional 3D micro and nanostructures through folding 2D substrates is that the fabrication of discrete devices (i.e. biological reactors, optical sensing, and digital electronic logic) is decoupled from the proper assembly process. The basic technologies which are required to manufacture 3D nanostructured components from 2D structures are developed and its feasibility is proven as shown above.

To expand the field of applications to more complex structures which meets future industry needs, the actual folding process as well as its preliminary definition of the folding structure (creases) needs to be explored. For this we will analyze first of all the modus operandi for a future case study of Nanostructured Origami™. Hereto we assume that the standard manufacturing process starts with an incoming order of a customer. From this point of view one can differentiate between two cases:

- a customer provides a final construction of a 3D shape with the order to fabricate this 3D shape from a 2D structure or
- the origami manufacturer is integrated into the development process of a new product and overtakes the task to assure that the desired 3D product is foldable from a 2D initial state.

Both cases implicate a different procedure. The first application requires an unfolding strategy. Here we draw the conclusion that if the 3D structure is unfoldable then the refolding from 2D into 3D is obvious. The second case implicates the inverse proceeding. The intention is to create a 2D crease structure and to prove if this structure is transferable into an appointed 3D shape. This approach represents a forward folding transformation from 2D into 3D.

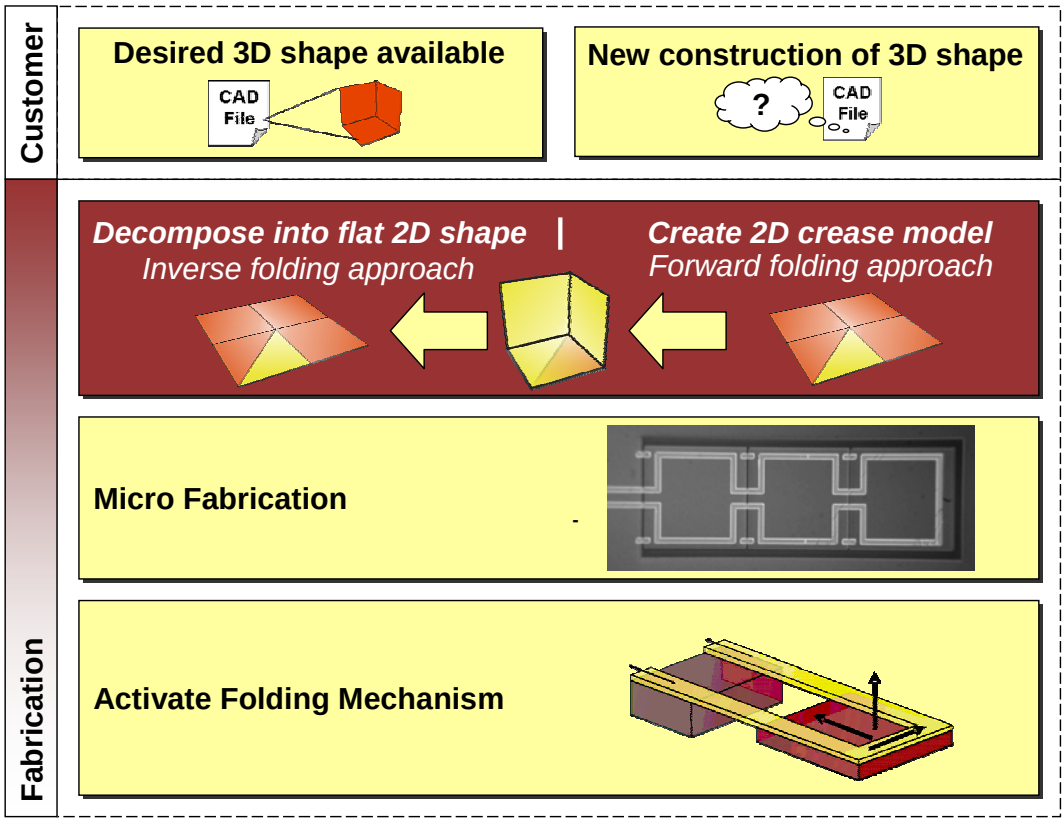


Figure 3-1: Conceptual formulation, two use cases

If both cases are satisfied, the above presented manufacturing and actuation methods are applicabe to complete the Nanostructured Origami^{TM} manufacturing process, see Figure 3-1.

Therefore there is an essential need to do research in the area of object folding to complete the NOFA process. Here we pursue the idea to determine the motion required to reach the goal from a given initial state by developing a mathematical notation which describes its kinematics. Further on we anticipate that the result of this work will be realized in a comprehensive software package which permits the easy, hands-on definition of foldable structures and their complete kinematic and dynamic analysis for design and verification in any given manufacturing challenge. To achieve this object the thesis is organized as follows, see Figure 3-2.

In the first place we present a general introduction to folding by first highlighting some general legality about origami and in particular about origami crease pattern. In addition the idea will be pursued to transfer the paper folding concept to a more generally admitted approach which uncouples the paper aspect from the folding operation. In chapter 4.2 the topological and kinematical classification of Multi-Body-Systems (MBS) will be pointed out and a well known mathematical formulation to describe the kinematical system will be established. In chapter 5 the folding and unfolding of

polygonal linkages are presented. Next a general procedure of how to solve the kinematic problem of a folding operation will be derived by considering the open corner demonstration model. The solution of the kinematic equations follows additionally with the object to compile the transfer functions for this example to permit a homogeneous animation of the folding process later on.

Further on we will shortly address the dynamics and self collision detection of folding objects. In connection to the folding approach we will outline the approach of Ileana Streinu (Smith College) concerning unfolding of polygonal linkages on the basis of a pseudo triangulation approach. Chapter 6 explains how the concepts and functional requirements developed in chapter 5 are transformed into general software architecture. The data management, visualization and functionality concepts are all addressed in detail. Chapter seven describes the realization of the above developed concepts. Last but not least we will illustrate the functionality of the developed software in terms of an example scenario. We conclude with a discussion of future work and applications, with special attention given to applications of NOS in future defense systems.

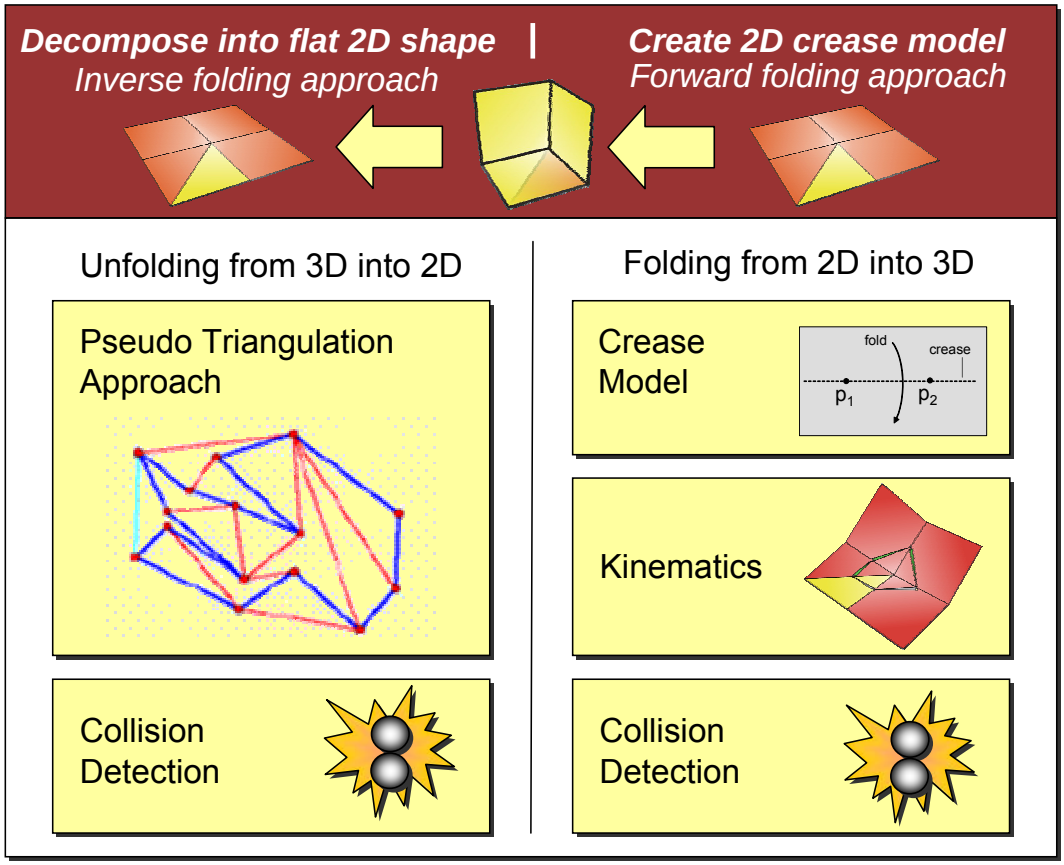


Figure 3-2: Use case analysis in detail

4 Analysis of Polygonal Linkages and Origami Structures

Folding is a very common process in our lives, ranging from the macroscopic level like paper folding or gift wrapping to the microscopic level like protein folding [AmSo 00]. Therefore, folding as well as the inverse proceeding unfolding are interesting research topics and have been studied in several domains of application. Determining the laws of folding – the “protein folding problem” – is one of the premier open questions in science. All fields of research have in common that they try to reach a desired final state, e.g., the gift package should be wrapped, or the protein's structure should be obtained; hence it is actually the knowledge of determining the motions required to reach the goal from a given initial state which must be understood.

For this reason, we believe that the use of origami structures in combination with perceptions in robotics and gearing technology has great potential to describe the kinematics of folding operations. The most attractive feature of origami is that one can construct a wide variety of complex shapes using a few axioms, simple fixed initial conditions and one mechanical operation, a fold. In the following chapter an introduction to the world of origami will be presented and some general legality will be pointed out. In addition the idea will be pursued to transfer the paper folding concept to a more generally admitted approach which uncouples the paper aspect from the folding operation. Hereto the analysis of polygonal linkages by consideration of its topological and kinematical classification will be presented.

4.1 Modelling Paper-Folding

The art and process of paper folding is called Origami. Origami comes from the two Japanese words ORI (to fold) and KAMI (paper) and was adopted into English as Origami [Mitc 03]. Origami is based on a modular conception which means that a number of individual "units," each folded from a single sheet of paper, are combined to form a compound structure [Weiss 99]. In order to describe an origami model and its geometric and combinatorial problems we have to establish first some terminology. To create a paper-folding model you have to think about the mechanics of putting creases into paper. To create a crease, in reality, you have to hold a fragment stationary, and fold the rest of the paper flat along the proposed crease line. Hereby we assume that our model has zero thickness and that our creases have no width and represent straight lines. The crease lines $l_1,...,l_n$ are enumerated in counter-

clockwise order beginning from the x-axis to describe the origami model. Each crease line is labeled with a pair of angles (α_i, Θ_i), where α denotes the position of the crease in the paper (the plane angle) and Θ denotes the folding angle. Further on each crease line can be distinguished into two types: *mountain creases* (M), which are convex, and *valley creases* (V), which are concave so each crease can be labeled as M or V [Hull 94]. To start the folding process (to be non flat), we change the dihedral angle δ of the crease to be greater than zero - or mathematically, we apply a rotation by $\pi - \delta = \Theta$.

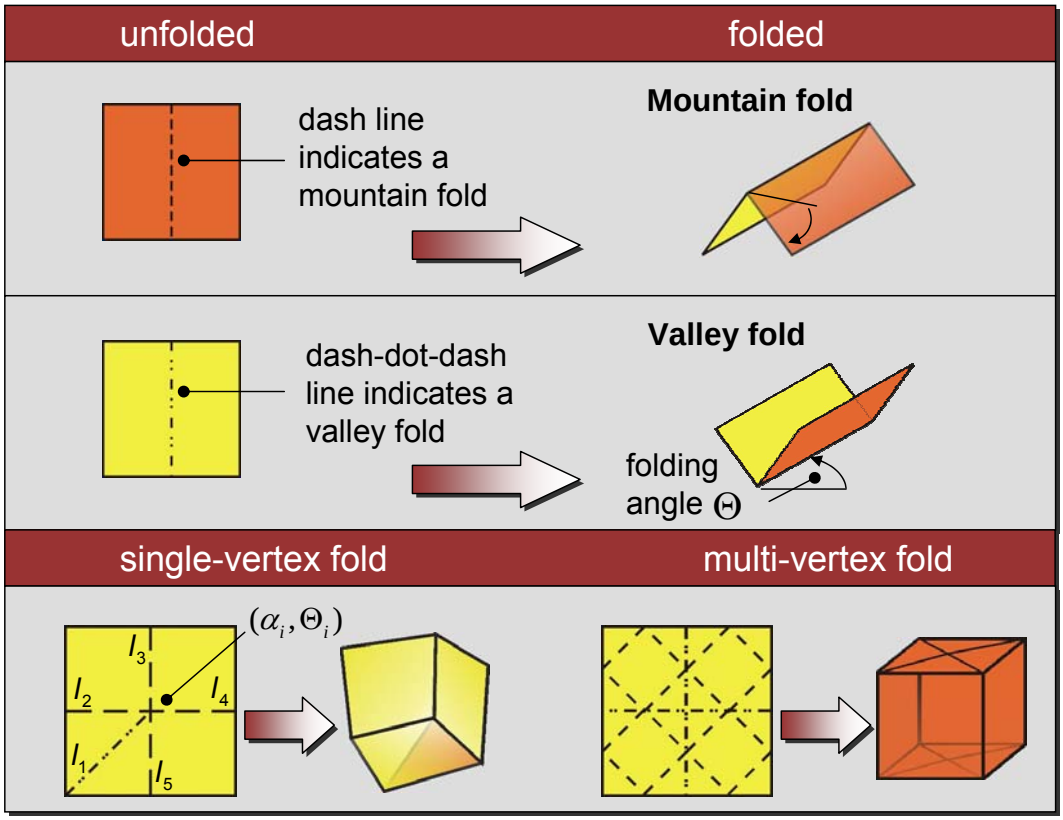


Figure 4-1: Basic origami terminology

Another important term of origami is the distinction between single-vertex and multi-vertex patterns. A single-vertex is characterized by the fact that all crease lines are incident to one point in the interior of the square. At this point it must be noted that all theorems which will be presented in addition are just applicable to single-vertex structures. Multi-vertex patterns present currently a large range of open questions to many origami mathematicians and there are no universal conditions available to describe these structures. Therefore, we will describe only single-vertex structures. But first of all, we will introduce a useful set of conditions to create crease patterns before we will start to analyze single-vertex patterns.

4.2 Huzita's Axiom

To create arbitrary crease structures, the Italian-Japanese mathematician Humiaki Huzita has formulated (1992) the most powerful known set of axioms related to origami crease patterns. These axioms do not describe all possible origami folds but its limits in describing 3D shapes are not known [Nagp 00]. Figure 4-2 shows axioms one to six with their corresponding parametric form:

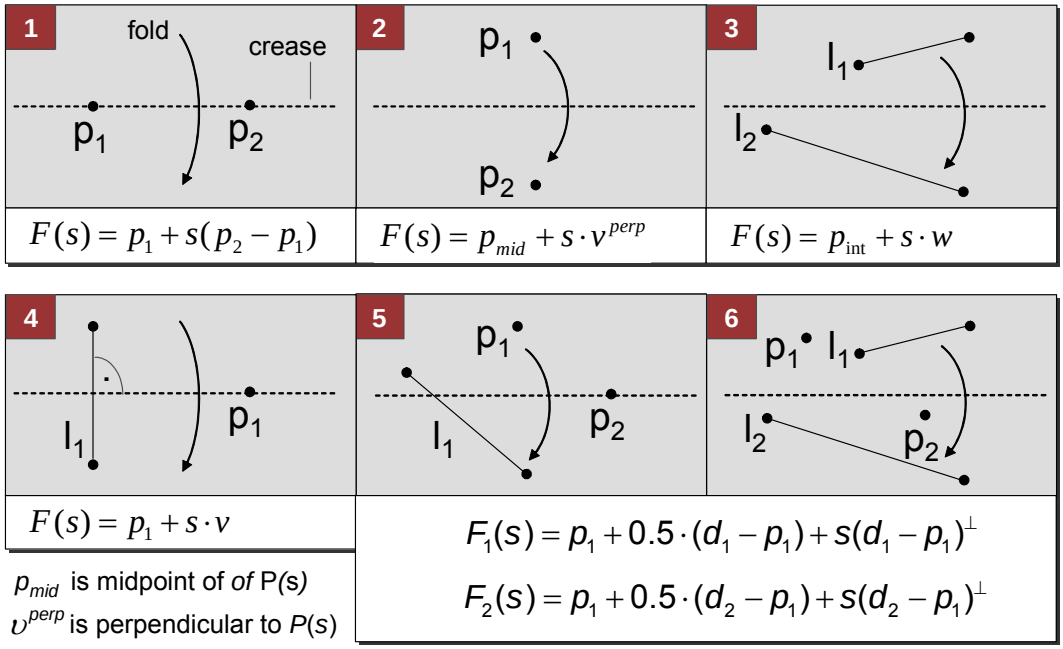


Figure 4-2: Huzita's axioms to model crease structures [Huzi 03]

Herein axioms 1 to 6 stand for in detail:

1. Given two points p_1 and p_2, there is a unique fold that passes through both of them.
2. Given two points p_1 and p_2, there is a unique fold that places p_1 onto p_2.
3. Given two lines l_1 and l_2, there is a unique fold that places l_1 onto l_2.
4. Given a point p_1 and a line l_1, there is a unique fold perpendicular to l_1 that passes through point p_1.
5. Given two points p_1 and p_2 and a line l_1, there is a fold that places p_1 onto l_1 and passes through p_2.
6. Given two points p_1 and p_2 and two lines l_1 and l_2, there is a fold that places p_1 onto l_1 and p_2 onto l_2.

We will refer to these axioms later on when the software architecture comes into play and a concept will be presented which describes how to create crease patterns.

Huzita has proven that the axioms can not only construct all plane Euclidean constructions, but also solve polynomials of degree three. Thereby some three dimensional origami models are featured with one specific characteristic; they are flat foldable – without adding or undoing any crease. In the following a set of theorems will be presented that provides necessary conditions for this property.

4.3 Analysis of a Single-Flat Vertex Fold

In order to be able to fold a paper model into the plane, we have to realize first the differences between flat-foldable and non-flat-foldable patterns. Hereby we can determine that not so much the number of folds as the number of vertices are the decisive factor. Flat fold means that the initial state of the folding model as well as its final shape can be represented in a 2D graph whereas a solid fold is characterized by its 3D shape in the final position. Kawasaki and Maekawa have done pioneering work in origami mathematics. They established two basic theorems relating to the single flat-vertex fold. Hereby the Kawasaki theorem gives a complete description of an alternating-sum condition that must be fulfilled to fold a single-vertex pattern locally flat [BeHu 02a].

Theorem 1 (Kawasaki)
Let ν be a vertex of degree 2n in single vertex and let $\alpha_1,...,\alpha_{2n}$ be the consecutive angles between the creases.
Then the creases adjacent to ν will (locally) fold flat if and only if

$$\alpha_1 - \alpha_2 + \alpha_3 - ... - \alpha_{2n} = 0$$

The requirement that the number of folds (2n) must be even follows from the fact that by an odd number of folds every time you fold cross a crease, a flat-folded model would flip out of the plane. If a single vertex-fold pattern satisfying Kawasaki's criterion, the next problem comes to bear to find the assignments of mountain and valley folds that achieve flat-foldability.
Here, Maekawa made the observation that the difference between the number of mountains and valleys at a single vertex (only) that folds flat is always two. This is known as Maekawa's Theorem. This means that if M-V=2 the vertex is pointing up and if M-V=-2 then it points down in its final position. By reversing all the mountains and all the valley creases you can flip a vertex pointing up to pointing down.

Theorem 2 (Maekawa)
Let M be the number of mountains and V be the number of valley creases adjacent to a vertex v in a single vertex fold.
Then $M - V = \pm 2$

But there are some constraints which mean that not all MV assignments for a flat vertex fold that satisfying $M - V = \pm 2$ are valid. If all angles are equal between the creases as in Figure 4-3(a), then any MV assignment which fulfills Theorem 2 is valid. But if we compare the crease structure of Figure 4-3(b) and Figure 4-3(c), we realize that if the crease lines l_1 and l_2 are both mountains or both valleys then the vertex will not fold flat.

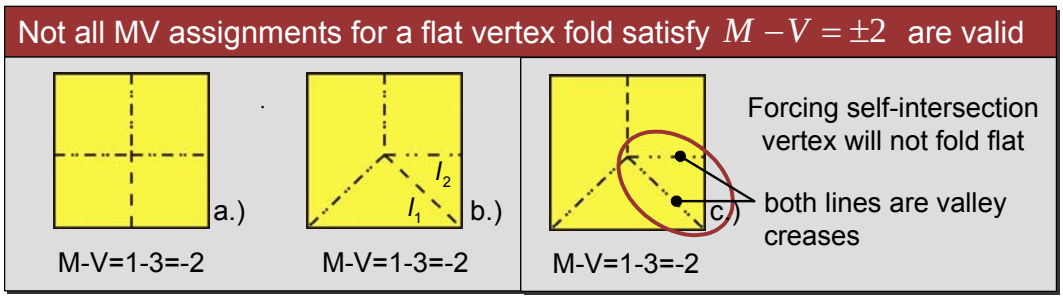


Figure 4-3: MV assignment

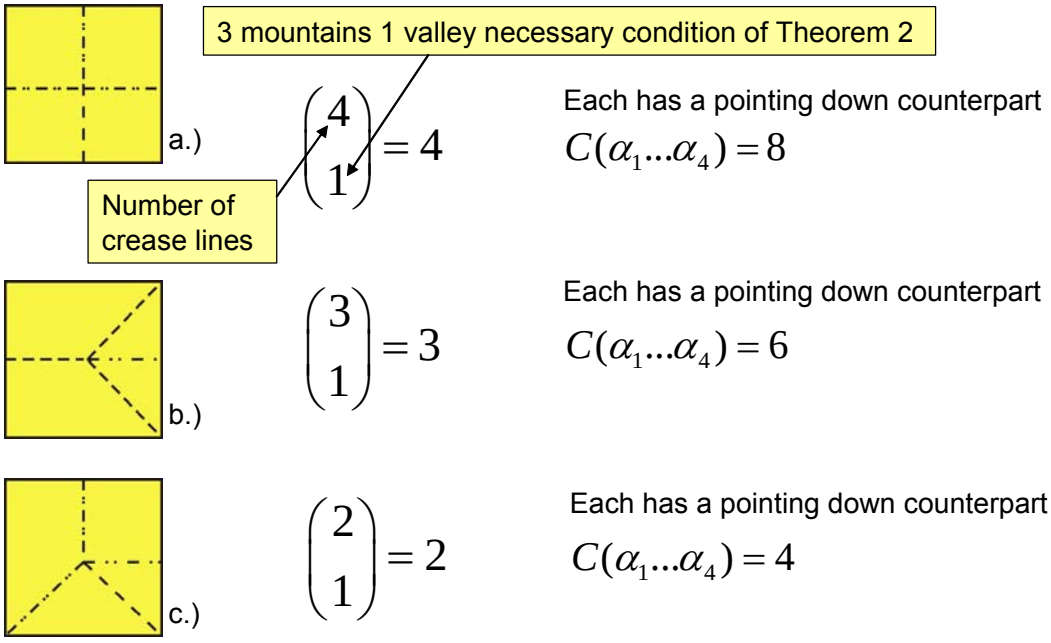


Figure 4-4: Number of possible MV assignments

This is would force a self intersection because the two neighboring right angles would try to cover the smaller 45° angle on the same side of the paper. This leads to the

case that line l1 and line l2 are either M,V or V,M creases. Maekawa's theorem then forces l3 and l4 to either be both M or both V creases. This results in two different possible MV assignments plus the pointing down counterparts, see Figure 4-3(c). This example points out that the number of valid MV assignments for a given flat vertex fold is depending on the allocation of the crease lines and the assignment of valley or mountain creases. Figure 4-4 shows the number of possible MV assignments for the left listed three different crease structures. In the first example there are four different assignments of one valley fold possible, so there are four different combinations available whereby each combination holds a pointing down counterpart. In the sum example (a) enfolded eight different MV assignments. Example (b) features six different combinations of MV assignments because of the fact that l1, l2, l3 must have two M's and one V, or vice-versa to fulfill Maekawa's theorem.

A number of researchers (Hull and Justin) have discovered a legality and formulated Theorem 3 which permits preconditioned that all angles around the vertex v are equal, in order to compute the number of valid MV assignments.

Theorem 3

Let $\nu = (\alpha_1,...,\alpha_{2n})$ *be the vertex in a flat vertex fold, on either a flat piece of paper or a cone. Then*

$$2^n \le C(\alpha_1,...,\alpha_{2n}) \le 2\binom{2n}{n-1}$$

are sharp bounds

Theorem 4

Let $\nu = (\alpha_1,...,\alpha_{2n})$ *be a flat vertex fold in either a piece of paper or a cone, and suppose we have* $\alpha_i = \alpha_{i+1} = \alpha_{i+2} = ... = \alpha_{i+k}$ *and* $\alpha_{i-1} > \alpha_i$ *and* $\alpha_{i+k+1} > \alpha_{1+k}$ *for some I and k. Then*

$$C(\alpha_1,...,\alpha_{2n}) = \binom{k+2}{\frac{k+2}{2}} C(\alpha_1,...,\alpha_{i-2},\alpha_{i-1}-\alpha_i+\alpha_{i+k+1},\alpha_{i+k+2},...,\alpha_{2n})$$

if k is even, and

$$C(\alpha_1,...,\alpha_{2n}) = \binom{k+2}{\frac{k+1}{2}} C(\alpha_1,...,\alpha_{i-1},\alpha_{i+k+1},...,\alpha_{2n})$$

if k is odd.

If the assignment of angles is not equal the equal-angle-in-row concept seems out of reach [see [Hull xx] for a full proof]. Hull has expanded Theorem 3 by using recursive formulae and created a universally valid theorem, see Theorem 4.

Theorem 4 provides a very efficient algorithm for computing $C(\alpha_1,...,\alpha_{2n})$ for any flat vertex fold. Considering the smallest angle, its neighbors will be larger than or equal to it. Here we will have an angle sequence satisfying the conditions of theorem 4. Repeat this with the new collection of angles, until all the angles are equal and theorem 3 can be applied [Hull xx]. Figure 4-5 shows an example of a single-vertex structure with six unequal angles. The application of Theorem 4 results in eight different possible MV assignments.

$$\mathrm{C}(100,70,50,40,30,70) = \binom{2}{1}\,\mathrm{C}(100,70,50,80)$$

$$= \binom{2}{1}\underbrace{\binom{2}{1}\mathrm{C}(100,100)}_{\text{Theorem 3}}$$

$$= \binom{2}{1}\binom{2}{1}2 = 8$$

Binomial Coefficient

$$\binom{n}{k} = \frac{n!}{(n-k)!k!}$$

Figure 4-5: Example how to compute number of valid MV assignments

The legalities and properties we have pointed out above reflect the large spectrum of the origami functionality. To put into a nutshell, Origami is an example of a language that constructively describes global structures - using a small set of axioms (Huzita's axioms) and only two types of folds (mountain and valley). One can construct a very wide variety of complex shapes and the initial conditions are very simple and always the same. Axioms generate new creases from existing points and creases and new points can be formed only by the intersection of previous folds. Origami can be characterized as a scale-independent language - i.e. the sequence of folds for a particular shape is independent of the size of the sheet [Nagp 00].

The application of these origami features and especially of the above presented mathematical achievements is ranging from safer airbags in cars to the development in outer space telescopes which benefit from this ease of transformation a large 3D shape into a compact 2D volume [CIP 03]. This property of specific structures to minimize their volume at a certain time and to extract their shape by unfolding is as well very interesting for nanostructured applications. But in the following we will focus on the counterpart, on solid single vertex structures.

4.4 Folding a Solid (non-flat) Single-Vertex Structure

As mentioned in chapter one, we are striving for an exactly constrained design that guarantees alignment tolerances of 10µm and does not require any special latching mechanism. Solid single-vertex structures are not flat foldable which means one can exploit this quality by designing structures remaining in a self latching position after the folding operation has been completed. In the following we will analyze a typical solid single-vertex model, namely an open corner or box pleated corner that rests in a self latching position after the folding operation. Here, we will investigate its folding properties taking into consideration its kinematic characteristics.

Henceforth in this thesis, we will make use of this optical element which reflects incoming light beams parallel to the angle it forays as a demonstration model.

In order to get familiar with the folding process and to receive an impression of the motion and relative motion of the single paper elements during the folding operation, a paper model is used to retrace the movement, see Figure 4-6. Noticeable in the context is that there are two independent folds necessary to achieve the desired 3D shape, see Figure 4-6. In a next step of development, the paper model is used to gain a better understanding of the perceived motion and it becomes obvious that a mathematical description of the kinematics is needed. Hereto we keep track of the idea to transfer the paper folding concept to a more generally admitted approach which uncouples the paper aspect from the folding operation. Therefore the idea will be pursued to use a combination of bodies, which are connected by joints, as a substituted mechanical system to describe the folding operation. For this purpose we will focus first of all on the analysis of Multi Body Systems (MBS).

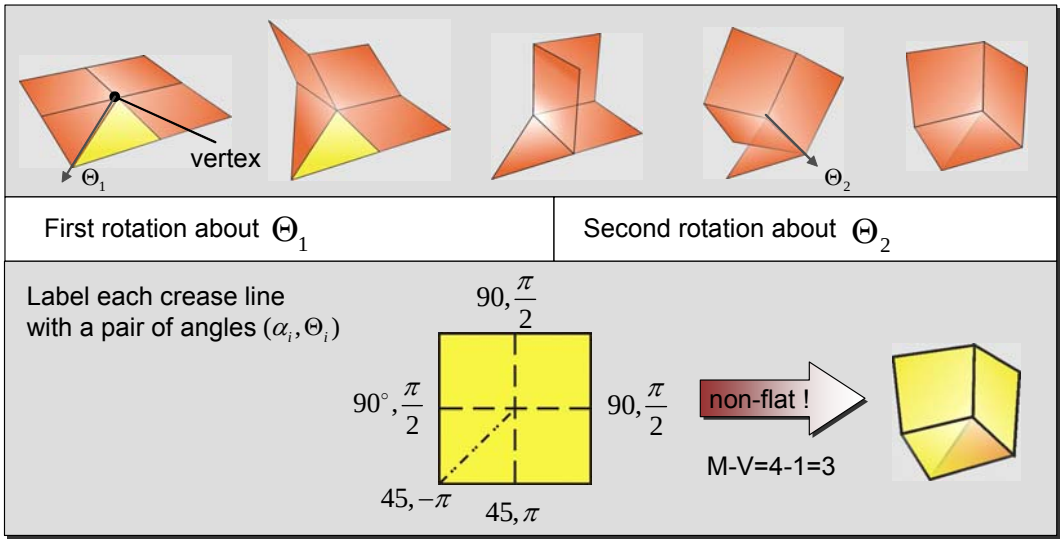


Figure 4-6: The corner cube as example for a solid single-vertex fold

4.5 Classification of Multi Body Systems

The use of MBS in terms of a substituted mechanical system is already applied in the area of animation software tools to describe the motion of body parts or in the field of robotics to describe the relative motion of robot arms. In general MBS are consisting of nB bodies, nJ joints and fJi degree of freedom and can be classified into two systems: Tree-Type Systems (TTS) and Multi Close Loop Systems (MCLS).

4.5.1 Topological Classification

In systems consisting of Tree-Type structures, the kinematical chain is featured that the way from one body to any other body of the system is explicit determined. Each body can be attached to a forerunner body resp. a forerunner joint. Thus, the relative motions between any two pairs of neighboring bodies are independent, and it is possible to process the kinetostatics of the elements on a component by component basis. Such a type of system is comparable with carton folding. If one folds a carton the relative motion between two pairs of carton are independent. Each piece of cartoon is featured with its own mobility, see Figure 4-7.

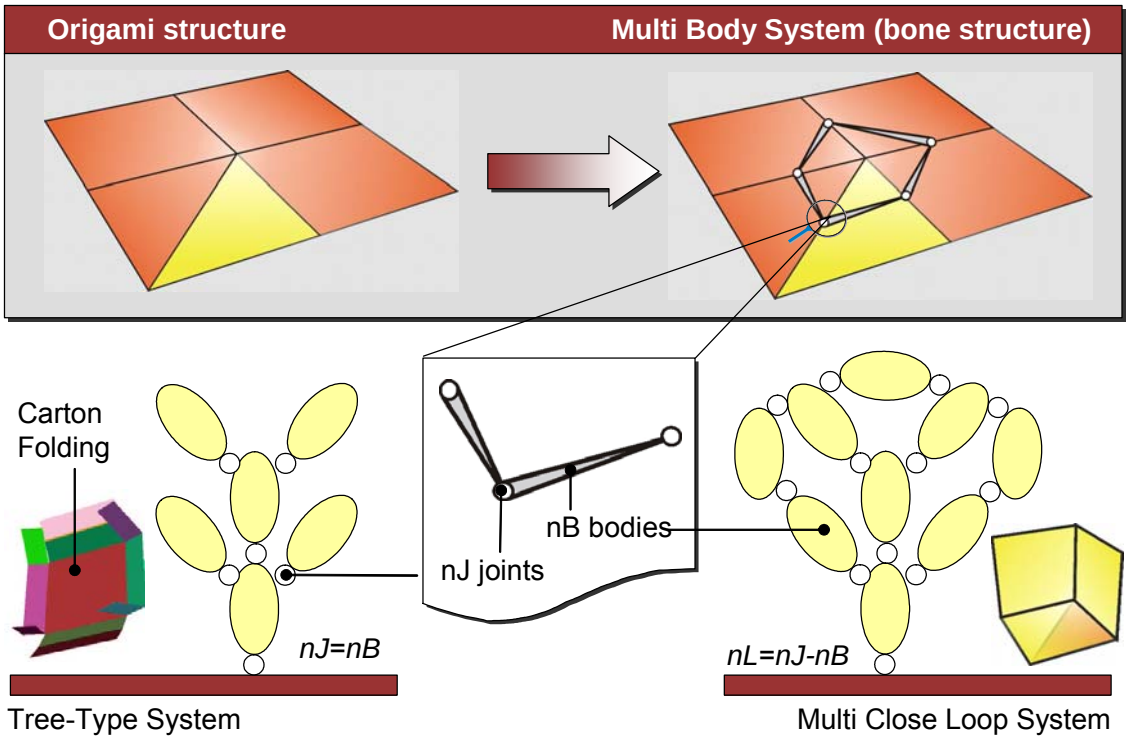


Figure 4-7: Topological classification of Multi Body Systems

On the contrary the open corner example we have introduced before, reflects a system of a close loop structure. An independent multi body system is generated by the insertion of an additional joint into a tree-type system.

Single or Multi Close Loop Systems are characterized by the fact that the relative motions within the loop become dependent; a change of relative motion at one location induces a change of relative motion elsewhere. This effect is noticeable during the most folding processes of origami models, as traceable in the open corner example, see Figure 4-6. Herewith the open corner representative for all single-vertex structures belongs to the topological classification of multi body systems. Next the kinematical classification of this model will be considered to constrict its kinematic motion.

4.5.2 Kinematical Classification

Multi Body Systems can be broken down into three groups depending on the motion of their bodies. The kinematic of our demonstration model can be attached to the motion of a spherical mechanism. Such a mechanism could be assembled of components which are cut out of a spherical surface whereby the axes of rotation are intersecting in the center of the sphere in one point, see Figure 4-8 [Diet 03].

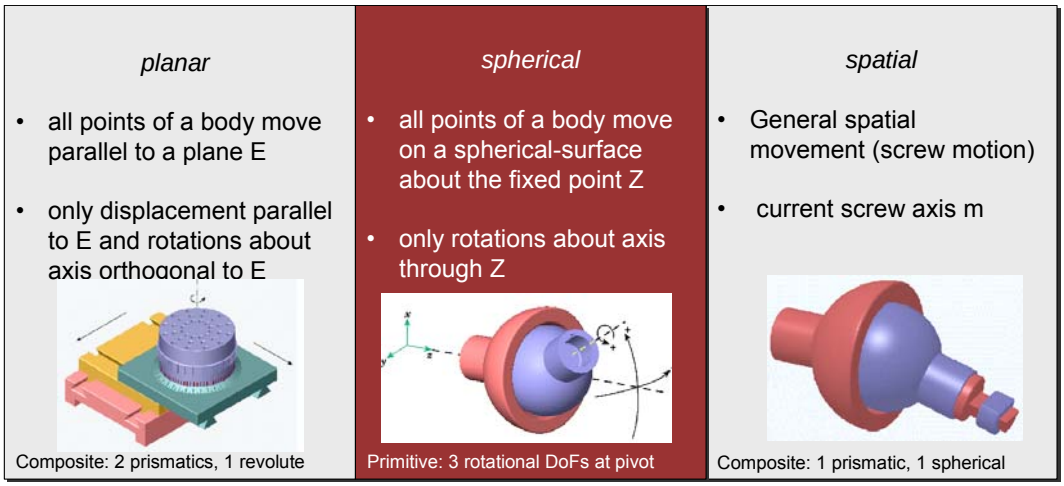


Figure 4-8: Kinematical classification of Multi Body Systems

In order to model this spherical mechanism, joints are necessary to connect two elements of a MBS with each other. In general, it establishes 6-f holonomic constraints between these two elements depending on the degree of freedom f. The relative motion of two bodies connected with a joint could be described with so-called natural or relative joint co-ordinates β. The natural joint co-ordinates are the angle of rotation $\beta_i = \Theta_i$ and/or displacement $\beta_i = s_i$. Figure 4-9 gives a review of the three basic joints which can be used to assemble joints with more than 1 DOF by series connecting of these swivel – and shear joints.

The most general joint is the helical joint (H). Special cases of this joint are the revolute joint (R) (flank lead=0) as well as the prismatic joint (P) (flank lead=1).

In order to create a substituted mechanical system which reflects the kinematical quality of an open corner we will model the creases as joints, and the uncreased regions as massless rigid bodies, see Figure 4-9. Hereto we will select revolute joints in combination with massless rigid bodies to simulate the spherical movement.

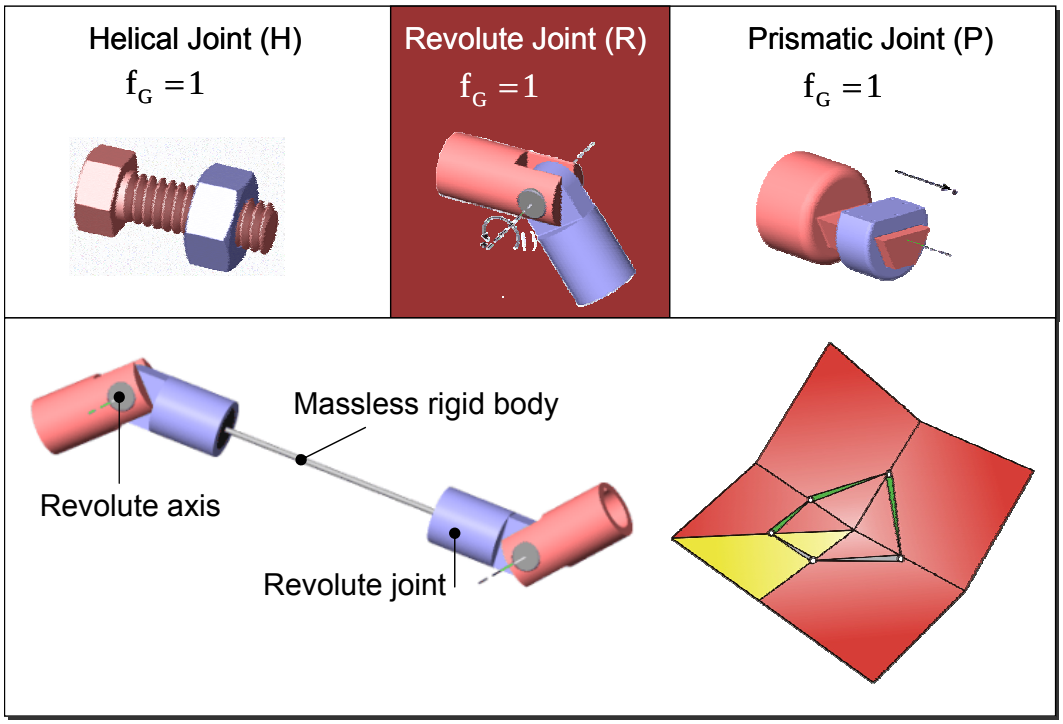


Figure 4-9: Necessary joint configuration to model an openCorner

The mobility of such a system represents a mechanism with two degrees of freedom (DoF). This means the object does not move uniquely from 2D into the final 3D shape. To move the substituted mechanical system in analogy to the paper folding model from the 2D initial state into the desired 3D shape we have to fix one body to the ground and lock one axis of rotation to create a constrained mechanism. This mechanism moves uniquely from 2D into 3D and is called a spherical 4R overconstrained mechanism [Diet 03]. After the first fold has been completed, the neighboring axis of rotation gets locked and the former locked axis gets twisted. With this strategy, locking one axis and getting a constrained mechanism with DOF equal to one, we can transfer the substituted mechanical system related to the paper model in two steps into the final 3D shape.

The characteristic of an overconstrained mechanism is mobility over a finite range of motion. Further on it must be mentioned that the Gruebner criteria with which it is possible to compute the mobility of general MBS can not applied to this kind of mechanism. In general one can compute the mobility of a system containing joints and with one link fixed to the ground with the Gruebner criteria whereby u_i represents the constraints and f_i the freedom of a joint:

$$\begin{aligned} M &= 6\cdot(n_B-1)-\sum u_i \qquad \text{with } u_i+f_i=6 \\ &= 6\cdot(n_B-1)-\sum 6-f_i \\ &= 6\cdot(n_B-j_i-1)+\sum f_i \qquad\qquad (1.3) \\ \\ &= 6\cdot(5-5-1)-\sum 6-1_i=-1 \end{aligned}$$

But there is a restriction of Gruebners formula if the mechanism is characterized as an overconstrained mechanism; equation (1.3) delivers a false result (-1) because particular connections are interdependent.

As a result the open corner represents a spherical mechanism which moves from 2D into a unique 3D shape by blocking respectively one axis of rotation. The axis of rotation can be modeled by revolving joints which are connected with each other by massless rigid bodies. The connection of these links represents a close loop multi body system whose kinematics will be analyzed additionally, and well-known mathematical methods will be applied to explicitly solve the kinematical equations.

4.6 Description of the Kinematics

The basic principle of solving kinematical systems is to reduce the degree of complexity to the minimum. In this case if we consider the folding operation we can determine that only a specific part of the complete system is moving, the other one is still not in motion. This is obvious because by locking one axis of rotation we transfer the system from a 5R overconstrained mechanism into a 4R overconstrained system. Therefore the backward link represents the immobile part.

4.6.1 Splitting the Mechanism into Two Systems

Now, the idea comes to bear to describe only the moving elements. Hereto one can break down the moving part of the close loop structure into two separate systems. The first one represents the twisted element which initializes the folding process by specifying the joint angles for each joint and delivers the position and orientation of

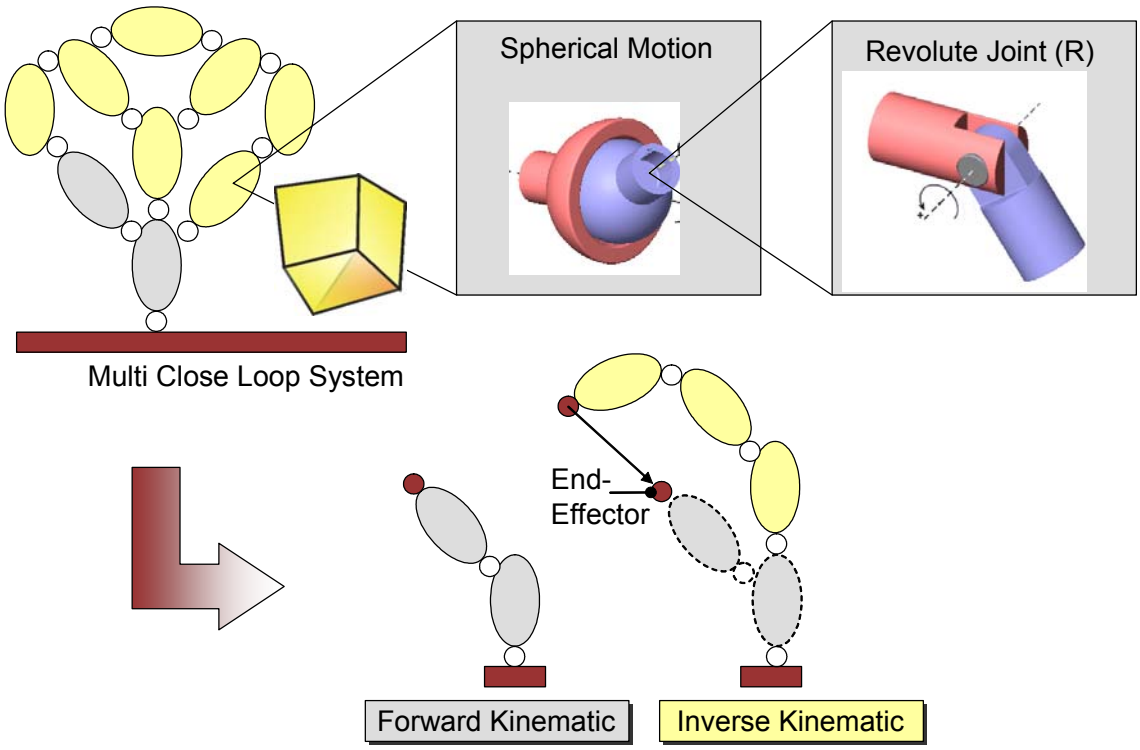


Figure 4-10: How to model the kinematics

the end effector, a classical Forward Kinematic (FK) approach. The second part tries to track the predetermined end effector position and represents an Inverse Kinematic (IK) problem which is well known and already established in robotics, see Figure 4-10. In a mathematical way, the inverse kinematic tries to find the joint vector q for a predefined position vector x (end effector) which fulfills the equation $q = f^{-1}(x)$. This approach represents the key feature of this thesis. Herewith the number of variables is reduced to a minimum because only participating elements of the motion are involved in the kinematical equations. The location of the coordinate system has an important bearing on the complexity of the kinematical equations, because the coordinates for the individual joints of a manipulator depend on the choice of the reference configuration [MrLS 94]. To simplify the equations as much as possible, we describe the two systems in two separate coordinate systems, which are so called “Frame A” for the forward kinematic part and “Frame B” for the inverse kinematic part, see Figure 4-11. The reference systems are located at the root of each kinematic chain. The computation of the position and orientation of the end effector represents a forward kinematic problem because it results in the multiplications of the transformation matrices of frame A and applying the corresponding twist angles. To solve the inverse kinematics is a much more complex problem. Next, different approaches to deal with inverse kinematic solutions will be presented.

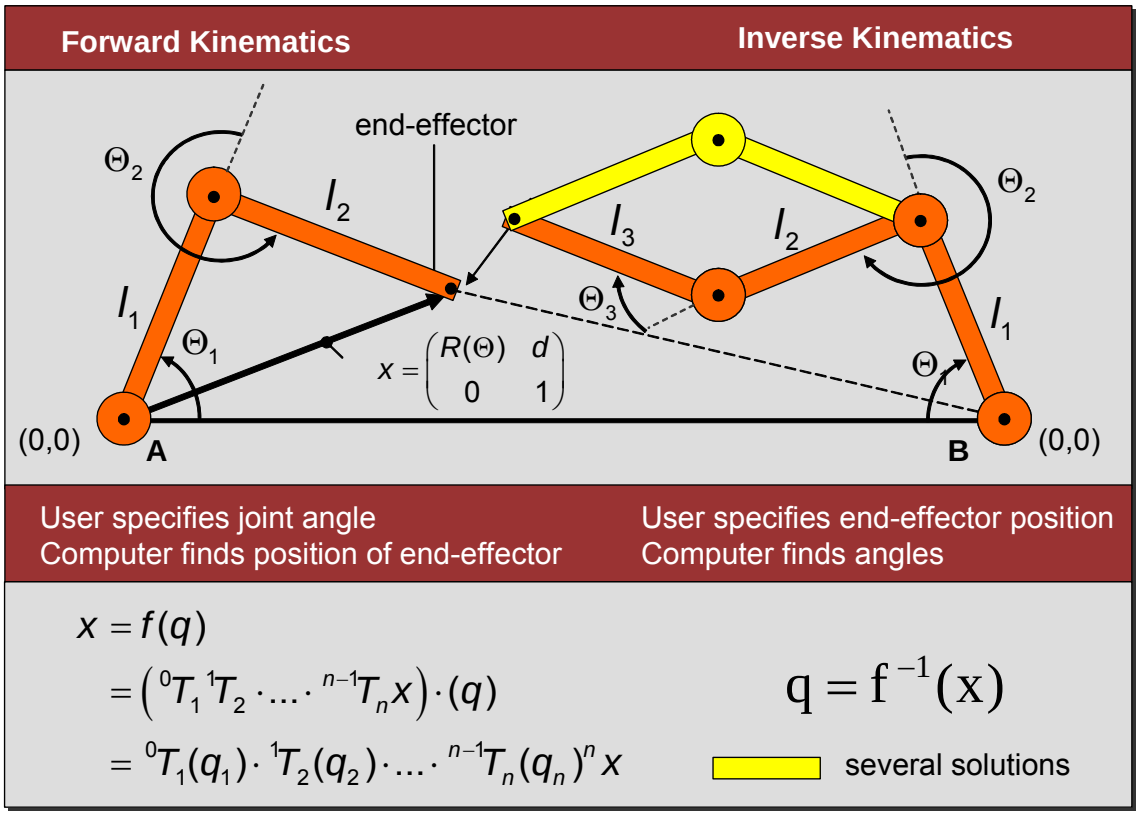


Figure 4-11: Splitting the system into a forward and inverse kinematic part

4.6.2 Solving the Inverse Kinematics

Inverse kinematics offers three ways how to get a solution. Possible methods are *algebraic, geometric and iterative.* Algebraic and geometric methods provide an exact solution and belong to the group of analytic solutions. If there are more solutions then these methods provide all possible solutions. If the end-effector is inaccessible then there is no solution available. Iterative methods offer a general solution of inverse kinematics and represent a numerical approach. Their disadvantage is that they converge to only one solution even if there are several ones or they find a closest solution if it doesn't exist [Kang 00].

Geometric methods use the knowledge of the manipulator geometry. This method bears the disadvantage that solutions for one manipulator cannot be transferred to a manipulator with different geometry. In order to solve inverse kinematics with algebraic methods we need to solve equations for N degrees of freedom. Every joint holds a transformation matrix M_i that consists of a translational and rotational part, both relative to its parents. The overall transformation is given by the product of the matrices along the path from the base frame coordinate system to the end effector by $M = M_i M_{i-1} \ldots M_2 M_1$.

With the use of this matrix multiplication M the forward kinematics can be computed in terms of the position and orientation of the end effector. The position and orientation of the end effector vector x in the parent's coordinate system is found by multiplying the position vector of the end effector q, in its own coordinate system with the transformation matrix M.

$$x = f(q) \tag{1.4}$$

In contrast to that, the inverse kinematics is needed to compute the state vector q for a given position and orientation of the end effector x which is the inverse function of (1.4)

$$q = f^{-1}(x) \tag{1.5}$$

The solution of (1.5) is not simple because the function f is not linear and consisting of terms of cosine and sine to describe the manipulator geometry. There is a one-valued function for (1.4) but for (1.5) there are more solutions q for one x as illustrated in Figure 4-11.

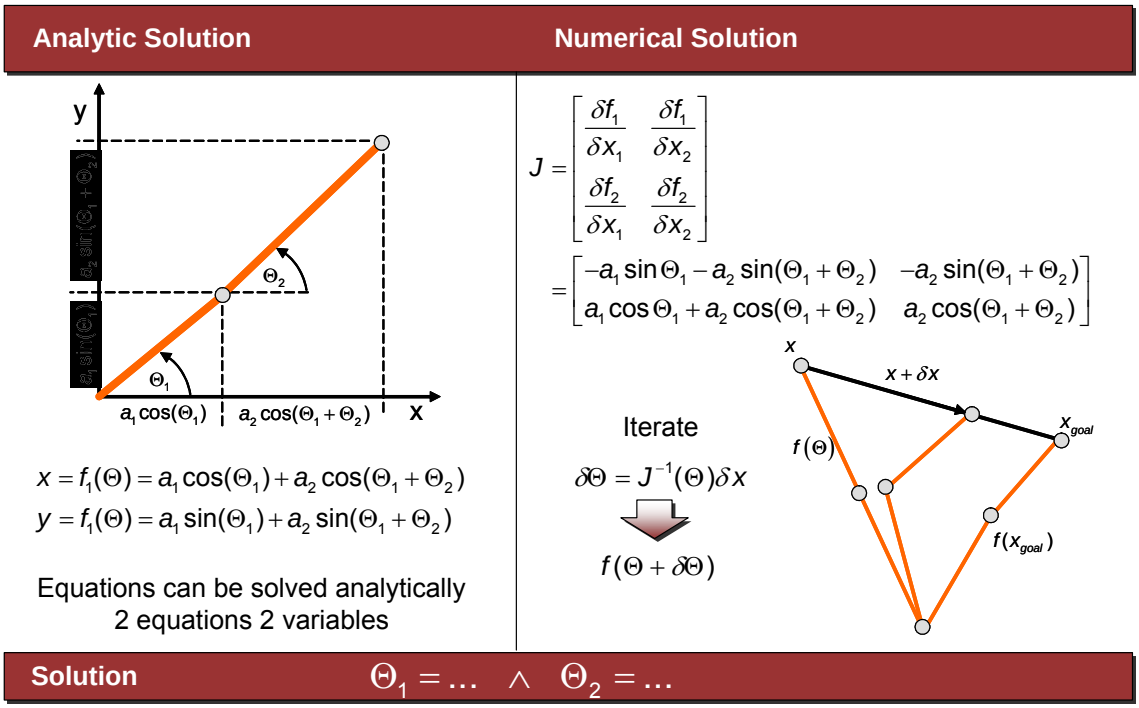


Figure 4-12: Two different solutions of the inverse kinematic equations

If the manipulator is changing its geometrical conformation, a numerical solution like the iterative approach offers a nice opportunity to compute its inverse kinematics. An Iterative solution is based on matrix inversion or on any form of optimization.

Matrix inversion is a complex process that is not only computationally difficult but presents additional problems due to numerical instability. Optimization methods bypass the problem of matrix inversion. These methods minimize the error in the kinematics chain. The Iterative approach is based on continuous closing to solution for every joint in a chain. In general these methods are more inaccurate and they converge only to one solution. But these methods are sufficiently good for kinematics simulation. Figure 4-12 shows an example of an iterative approach on the basis of the matrix inversion of the Jacobi matrix.
Nevertheless we will focus on an analytic solution because the geometry of our substituted mechanical system can be extracted from the predefined crease structure which is well known. In a next step a mathematical formulation is needed to realize the kinematic configuration.

4.7 Mathematical Formulation

In this section, a geometric view helpful to understand translational and rotational motion of rigid bodies will be presented. Different methods have been developed to describe kinematic chains, mechanisms and manipulators in the past. Primarily the methods differ in the description of the rotation of a rigid body because the description of the translatory movement does not represent a difficulty anyway. In order to state only a few methods, beside the simple geometrical methods, the vector analysis, the matrix formulation, and the quaternion computation are established approaches to describe kinematical conditions. In the following we will focus on a well-known mathematical method called matrix exponential formulation (MEF) to describe the above established forward and inverse kinematic systems.

4.7.1 Screw Calculus

The elements of screw theory can be traced back to the work of Chasles and Poinsot in the early 1800s. Chasles proved that a rigid body can be moved from any position to any other by rotational movement around an axis in space through the angle of radians, followed by translation along the same axis. This motion is what we consider in the following as a screw motion [MrLS 94].
The infinitesimal version of a screw motion is a twist. It provides a description of the instantaneous velocity of a rigid body in terms of its linear and angular components.
The second major result upon which screw theory is founded upon concerns the representation of forces acting on a rigid body.

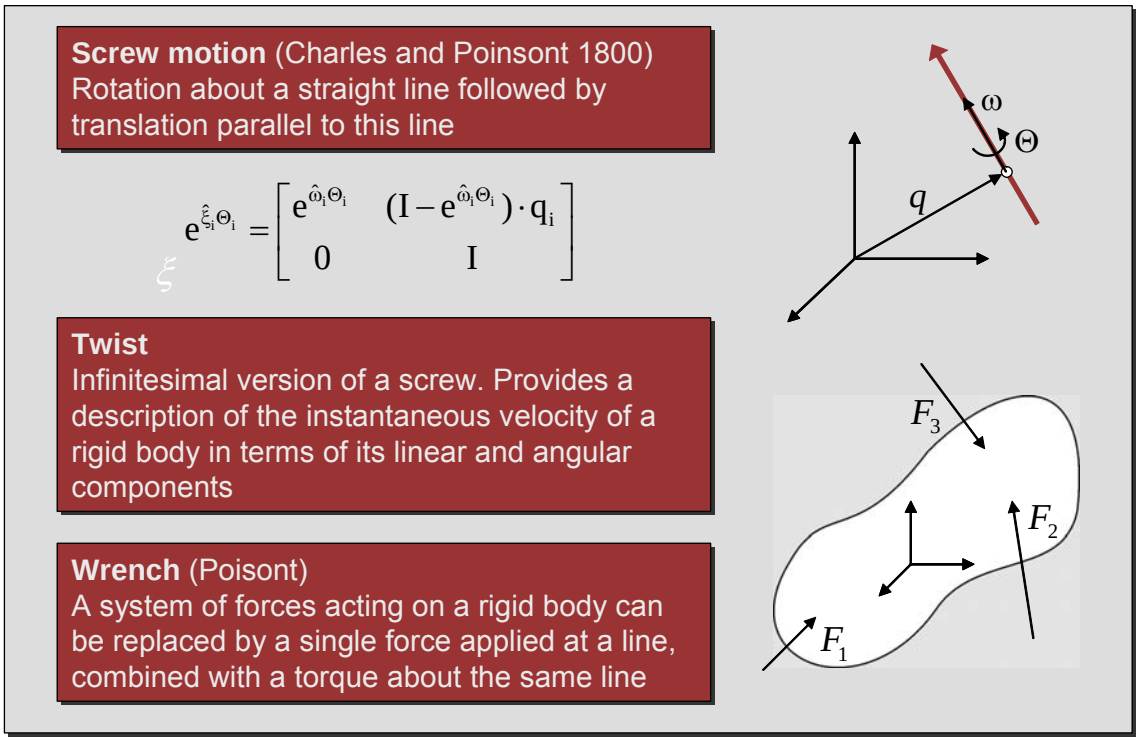


Figure 4-13: Screw Calculus in detail

Poinsot is credited with the discovery that any system of forces acting on a rigid body can be replaced by a single force applied to a line, combined with a torque about the same line. Such a force is called a wrench.

In the next paragraphs we will introduce the notion of screws and twists, and describe their relationship with homogeneous transformations. In the following chapter we will show that the kinematics of close-link manipulators can be separated into a forward and inverse kinematic part represented as a product of exponentials. This formulation is first pointed out by Brocket, is elegant and this treatment of kinematics is something of a deviation from most other approaches, which prefer a Denavit-Hartenberg formulation of kinematics. There are two main advantages of using screws, twists and wrenches for describing rigid body kinematics.

- They allow a global description of rigid body motion which does not suffer from singularities due to the use of local co-ordinates.
- The screw theory provides a very geometric description of rigid motion which greatly simplifies the analysis of mechanisms

5 Folding of Polygonal Linkages

All requirements to start describing the folding of polygonal linkages are set. The topological and kinematical classification is presented as well as a mathematical formulation in terms of the exponential matrix formulations is defined. In a next step the above developed concept, to apply a substituted mechanical system to a paper folding model, will be reconstructed in detail on a demonstration model.

5.1 The Open Corner Demonstration Model

Initially, the crease structure of the model must be described. The pattern of an open corner consists of five creases which are arranged around the vertex in the interior of the paper plane. The two neighboring angles α_1 and α_2 enclose in each case an angle of 45° whereby the other three ($\alpha_3, \alpha_4, \alpha_5$) include an angel of 90°. Further on the model comprises four valley creases and one mountain crease to be foldable from 2D into the desired 3D shape. Hereby the mountain crease is applied to the second axis of rotation. That must be considered if one determines the orientation of the rotational axis in detail. According to this, the other creases represent valley creases, see Figure 5-1.

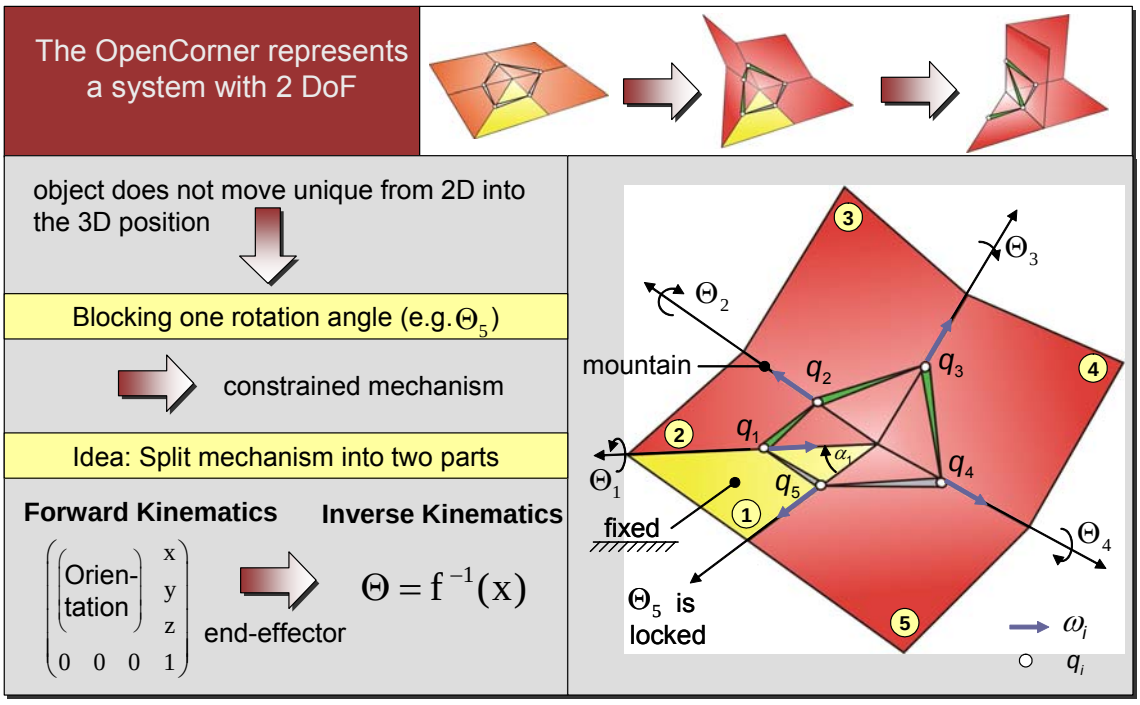


Figure 5-1: Kinematic configuration of the open corner demonstration model

After the crease structure is defined, the substituted mechanical system can be applied. As mentioned in chapter 4 the spherical movement will be modeled by revolute joints which are connected with rigid bodies around the vertex. Therefore each crease line will be replaced with a revolute joint consisting of a line item specification in respect to its corresponding reference system and an orientation with regard to the rotational direction (mountain or valley crease). Among each other, all joints are connected by bodies to form a close loop structure, see Figure 5-1.

Transferred to the exponential matrix notation this means that each joint is characterized by the point q_n, where the axis of rotation is going through and the unit vector ω_i in the direction of the twist axis. Hereby the joint vectors q_n are pointing from the corresponding reference system to the joints. In the first instance, all joint vectors will be described in reference to a base co-ordinate system in the interior of the plane. If all creases are replaced by joints and described with the use of the two parameters q_n and ω_i, the actual description of the folding operation can be as follows.

Similar to the paper folding model, where one holds a fragment stationary, and fold the rest of the paper flat along the proposed crease line, one body (1) will be connected to the ground. Hereupon two possible axes of rotation (Θ_1, Θ_2) to initialize the folding operation come to bear, see Figure 5-1.

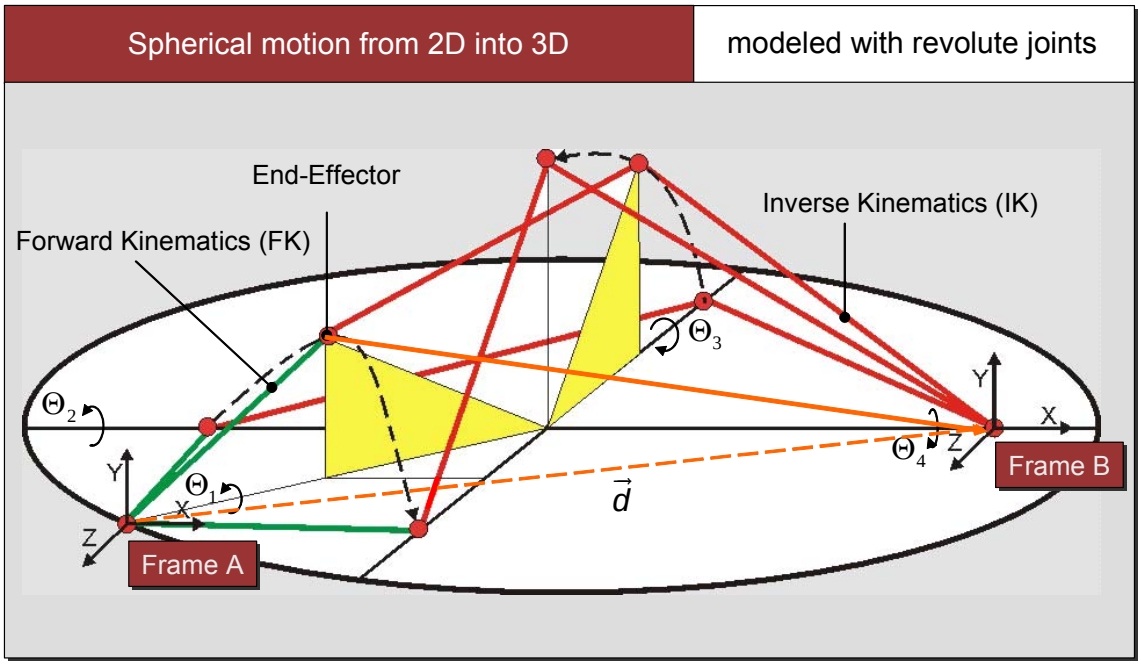


Figure 5-2: Illustration of a 4R overconstrained mechanism

As mentioned in chapter 4 the open corner model moves only from its 2D initial state into a unique position in 3D if the DoF of this mechanism is equal to one. This necessary condition will be satisfied by locking one axis of rotation. Hereto we will freeze axis Θ_2 and transfer the spherical mechanism into a 4R overconstrained mechanism which can be moved into a constrained position by twisting Θ_1.

On going we can split the mechanism into two sub-systems, one forward kinematic and one inverse kinematic system. The forward kinematic chain describes the line section from point q_1 to q_2. According to this, the inverse kinematic chain comprises the line section q_4 to q_2. Hereto the coordinates of the joint vectors q_i need to be transferred to their new reference systems “Frame A” resp. “Frame B”, see Figure 5-3. The point q_2 is represented in both systems and is called the end-effector. Figure 5-2 illustrates this spherical motion if the twist angle Θ_1 reaches a value of 0° to 180°. In this example q_2 describes a curve which represents the run of the objective function of the inverse kinematics. The position and orientation of each point on this curve can be expressed in frame B by including the distance vector $\vec{d}$. In combination with the line item specification of q_2 in reference to “Frame A” one can compute the position and orientation of the end effector according to Frame B by $\vec{q_2} - \vec{d}$. This result represents the input parameter of the inverse kinematic system.

Mathematical Description: Exponential Matrix Formulation	$\Theta = g_{IK}^{-1}(g_{FK}(\Theta_1))$

Θ_n: angle of twist
$e^{\hat{\xi}\Theta}$: exponential coordinates for rotation
$\longrightarrow$: axis of rotation
q_n: points, where the axes of rotations are going through
ξ_i: twist corresponds to the screw motion for the i[th] joint
ω_i: unit vector in the direction of the twist axis

FK	$g_{st}(\Theta) = e^{\hat{\xi}\Theta_1} \cdot e^{\hat{\xi}\Theta_2} \cdot g_{st}(0)$
IK	$g_{st}(\Theta) = e^{\hat{\xi}\Theta_4} \cdot e^{\hat{\xi}\Theta_3} \cdot e^{\hat{\xi}\Theta_2} \cdot g_{st}(0)$

Frame A: $q_1 = \begin{pmatrix}0\\0\\0\end{pmatrix}$ $q_2 = \begin{pmatrix}0\\1\\0\end{pmatrix}$ $\omega_1 = \begin{pmatrix}1\\1\\0\end{pmatrix}$ $\omega_2 = \begin{pmatrix}1\\0\\0\end{pmatrix}$

Frame B: $q_4 = \begin{pmatrix}0\\0\\0\end{pmatrix}$ $q_3 = \begin{pmatrix}-1\\1\\0\end{pmatrix}$ $q_2 = \begin{pmatrix}0\\-2\\0\end{pmatrix}$ $\omega_4 = \begin{pmatrix}1\\0\\0\end{pmatrix}$ $\omega_4 = \begin{pmatrix}0\\1\\0\end{pmatrix}$ $\omega_2 = \begin{pmatrix}1\\0\\0\end{pmatrix}$

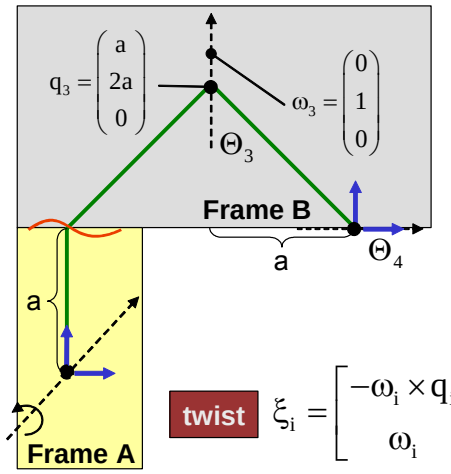
$$e^{\hat{\omega}\Theta} = I + \hat{\omega}\sin\Theta + \hat{\omega}^2(1-\cos\Theta)$$

$$e^{\hat{\xi}_i\Theta_i} = \begin{bmatrix} e^{\hat{\omega}_i\Theta_i} & (I - e^{\hat{\omega}_i\Theta_i}) \cdot q_i \\ 0 & I \end{bmatrix}$$

Figure 5-3: Application of Screw Calculus theory

With the use of q_1 to q_5 and ω_1 to ω_5 the forward and inverse kinematic equations are almost set. To complete each equation, the transformations between the end effector and the base frames A and B at $\Theta = 0$ stand open. This connection will be expressed in terms of $g_{st}(0) = \begin{bmatrix} I & (x \quad y \quad z)^T \\ o & 1 \end{bmatrix}$ for the FK as well as the IK system of equation.

After the equations to compute the folding operation over Θ_1 are set, we will execute the computation for the twist angle $\Theta_1 = 90°$ in detail. The end effector position results to $\begin{pmatrix} 0.5 & 0.5 & 1/\sqrt{2} \end{pmatrix}^T$ by computing the forward kinematic (FK) equation, see Figure 5-4. Expressed in frame B using the distance vector $\vec{d} = \begin{pmatrix} 2 & 1 & 0 \end{pmatrix}^T$ the position results to $(-1.5 \quad -0.5 \quad 1/\sqrt{2})^T$ and delivers the input parameter for the inverse kinematics equation (IK). The solution of the inverse kinematics comprises a non linear system of equations. The information about orientation and position of the end effector delivers together 9+3=12 equations to solve the inverse kinematics. This over determined system can be solved analytically. As a result we will receive two pairs of angles: $(\Theta_3 = 60° \wedge \Theta_4 = 35.5°) \vee (\Theta_3 = -60° \wedge \Theta_4 = -35.5°)$. In this case the predefined crease structure delivers one constrained condition;



twist $\xi_i = \begin{bmatrix} -\omega_i \times q_i \\ \omega_i \end{bmatrix}$

$$e^{\hat{\omega}\Theta} = I + \hat{\omega}\sin\Theta + \hat{\omega}^2(1-\cos\Theta)$$

screw $e^{\hat{\xi}_i\Theta_i} = \begin{bmatrix} e^{\hat{\omega}_i\Theta_i} & (I - e^{\hat{\omega}_i\Theta_i}) \cdot q_i \\ 0 & I \end{bmatrix}$

FK $g_{FK}(\Theta) = e^{\hat{\xi}\Theta_1} \cdot e^{\hat{\xi}\Theta_2} \cdot g_{st}(0)$

$$g_{FK}(\Theta_1 = 90°) = e^{\hat{\xi}\Theta_1} \cdot e^{\hat{\xi}\Theta_2} \cdot g_{st}(0) = \begin{pmatrix} 0.5 & 0.5 & 1/\sqrt{2} \end{pmatrix}^T$$

IK $g_{IK}(\Theta) = e^{\hat{\xi}\Theta_4} \cdot e^{\hat{\xi}\Theta_3} \cdot e^{\hat{\xi}\Theta_2} \cdot g_{st}(0)$

$$= \begin{pmatrix} C\Theta_3 & 0 & S\Theta_3 & -C\Theta_3 - 1 \\ S\Theta_4 S\Theta_3 & C\Theta_4 & -S\Theta_4 C\Theta_3 & -S\Theta_4 S\Theta_3 \\ -C\Theta_4 S\Theta_3 & S\Theta_4 & C\Theta_4 C\Theta_3 & C\Theta_4 S\Theta_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\Theta = g_{IK}^{-1}(g_{FK}(\Theta_1 = 90°))$$

Figure 5-4: Kinematic configuration in detail

Here the orientation of ω_2 plays the decisive role which leads to the result that only the execution of a mountain fold is admitted and a rotation about a negative angle would break this condition. Therefore the negative angle solution can be canceled.

Curve Fitting

On going the folding operation for $\Theta_1 = 90°$ can be retraced by applying values $\Theta_3 = 60° \wedge \Theta_4 = 35.5°$ to the corresponding axis of rotation. In the view of developing a simulation software tool which should animate the folding operation, the solving of the inverse kinematics for each small angle position between 0° and 180° is computationally extensive. Hence we can compute the corresponding angles in large steps of, e.g., 10° one-time and apply a curve fitting method to reconstruct the motion at intermediate positions, see Figure 5-5. The polynomials found through curve fitting describe the folding operation for arbitrary angle positions in the range of 0° to 180°. The degree of the polynomial as well as the number of sampling points defines the accuracy of the fitting. With the use of these transfer functions, describing the relation between the twisting angle and its constraint angles, the time-critical aspect of the animation is avoided.

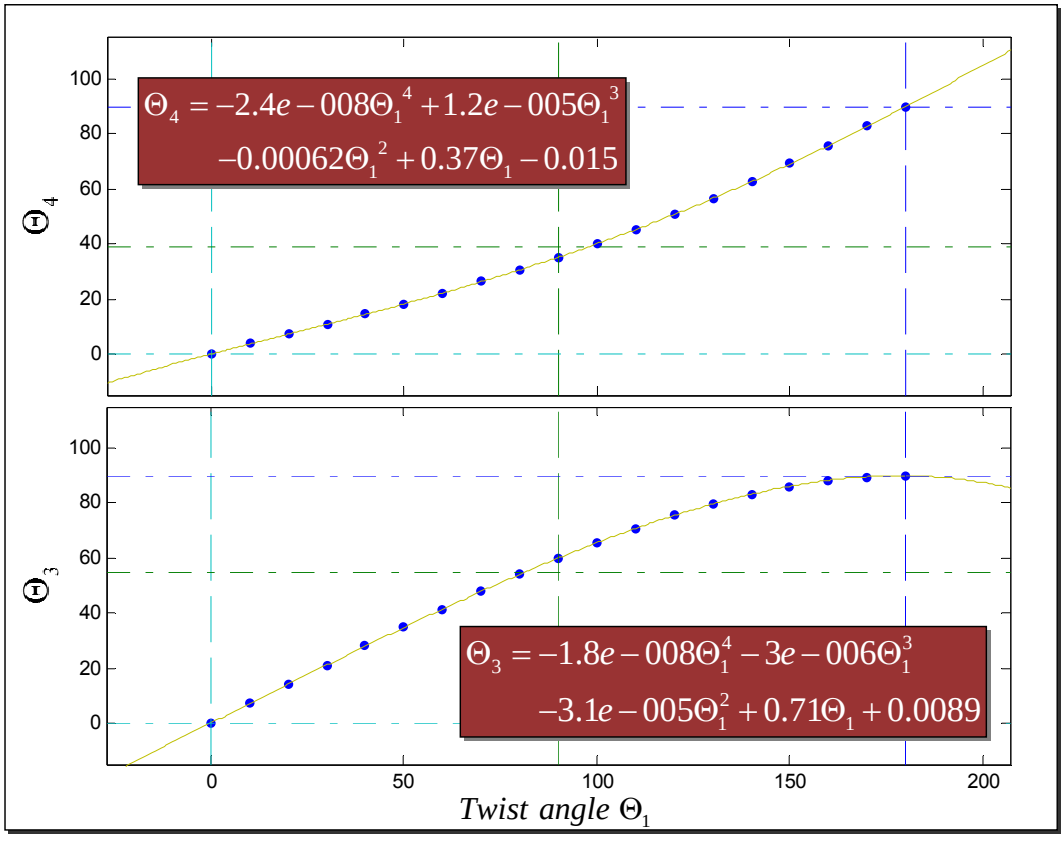


Figure 5-5: Polynomial Curve Fitting: 4th degree

Second Fold

Up to now the open corner does not reach its final 3D shape, consequently a second fold is necessary to complete the folding. Hereto we will lock Θ_1 and Θ_5 becomes the new twist angle. But in order to describe the second fold, first of all, we have to determine the current coordinates of q_1 to q_5 in space after the first fold has been completed. Hereto each q_i will be multiplied with a corresponding rotation matrix. At this juncture, the sequence of multiplication must be kept in mind. The computation of the position of q_2 in reference to frame B, e.g., results in $q_2' = R(\Theta_3) \cdot R(\Theta_4) \cdot q_2$.

After the present positions of q_i in reference to “Frame A” resp. “Frame B” have been established, the close loop mechanism can be re-split into a new forward and inverse kinematic part. Now the line section q_5 to q_4 in reference to Frame A’ builds up the forward kinematic and the line section q_2 to q_4 in reference to Frame B’ forms the new inverse kinematic chain, see Figure 5-6. Before one can start to draw the system of equations for the second fold all joint coordinates q_i need to be transformed into the new frames A’ and B’.

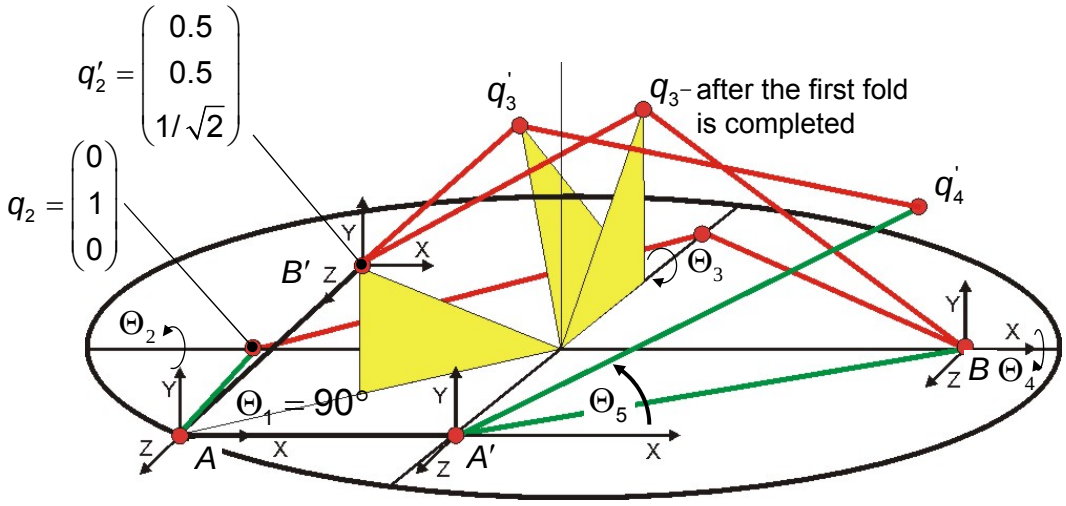


Figure 5-6: Second fold about Θ_5

Hereupon the proceeding to calculate the corresponding angles for a predefined twist angle and applying a curve fitting through some significant points as presented above for the first rotation about Θ_1 can be executed. With the combination of these two folds the open corner will be transferred from its 2D initial state into the final 3Dhape.

5.2 Polynomial Linkages

This approach has great potential to describe the folding behavior of arbitrary vertex structures. Even more complex patterns with more than one vertex might be able to be simulated by connecting several single vertex structures to a multi vertex model. The element which connects two vertices with each other would represent the output parameter for the forerunner as well as the input parameter for the back runner model. This forerunner, back runner principle permits to describe the folding operation of even complex structures. Further on this approach offers the opportunity to analyze folding structures taking into consideration the foldability by analyzing the result of the inverse kinematics. If the IK does not supply any valid angular position the crease pattern is invalid in this current composition and a revision of the crease structure is necessary. In the following we will shortly address the advantage of the MEF concerning the implications of the dynamics of folding objects.

5.3 Dynamics of Folding Objects

The Screw matrix formulation has the additional advantage that it permits the simulation of the folding structure's dynamic properties. This is done as follows: based on the twist/screw matrices, one can define a complementary dual set of reciprocal screw and wrench matrices [MrLS 94], which describes the compliance and dissipation of the hinges or deformable crease interfaces between segments. The matrix elements can include non-linear forces as well to model plastic deformations which are of particular interest in the future. The wrench formulation is beyond the scope of this thesis, but the approach set herein permits its straightforward future inclusion.

5.4 Self Collision Detection of Folding Objects

Kinematics compatibility (or collision avoidance): Do different hybrid objects, mounted on the base frame and cooperating for a special purpose in the final 3D shape, collide with each other during the folding process? We pursue two methods to monitor collisions: one "brute force" method where at each time step we solve for intersections of the folding segments based on their positions, as given by the product-of-matrix-exponentials formula; and an approximate method with which we define sample points on the segments and, at each time step, we detect if these points get within a "cloud of proximity" from points of other segments.

6 Unfolding of polygonal Linkages

At this point the folding of polygonal linkages has been considered and a concept to describe the kinematics of multi linkages has been developed. Further on, the second categorical application, the unfolding of polygonal linkages, will be shortly addressed whereby a full description is not within the scope of this thesis.
In general the unfolding represents the inverse case of folding. Here the initial condition comprises a 3D folded shape which should be unfolded (expanded) without any collision. The problems of unfolding have been implicit since Albrecht Dürer in the early 1500's [Dema 03]. Over the past few years, there has been a surge of interest in these problems in discrete and computational geometry. The work of Connelly, Demaine, and Röte was a breakthrough in this field - relatively shortly after their work had appeared, Ileana Streinu (Smith College) provided a dramatic new approach to solve the geometrical challenge of unfolding polygonal linkages.
The main idea of her approach is illustrated by a motion-planning problem: how to continuously reconfigure a simple planar polygon to any other planar configuration with the same edge-length while remaining in the plane and without creating self intersections along the way. This means, e.g., a simple (open or closed) planar robot arm can be unfolded to a convex position (and hence reconfigured to any other similarly oriented position) with a sequence of at most $O(n^2)$ non self-intersecting, expansive motions induced by one degree of freedom mechanisms defined from pseudo triangulations [Stre 03]. In the following the fundamental principle of her approach will presented.

6.1 Pseudo Triangulation Approach

To solve the unfolding problem, we start with a folded origami structure, and project the skeleton on a plane coinciding with the plane where the structure will eventually be unfolded, see Figure 6-1(b). This projection is a combinatorial graph G=(V,E) embedded in the plane with rigid bars (fixed length straight line segments) corresponding to the edges. Planning the linkage motions requires very complex strategies for simultaneously controlling all joints and avoiding collisions. At this point the idea comes to bear to decompose the trajectory into simple motions using one degree of freedom mechanisms.

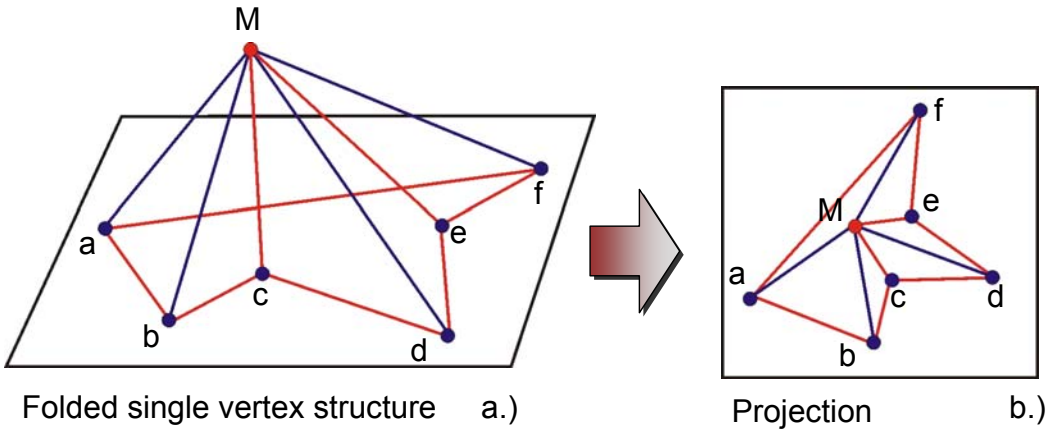


Figure 6-1: Collecting the geometrical shape [Stre 03]

Hereto, extra bars will be placed in such a way that they “almost” form a pseudo-triangulation guarantee that the motion has only 1 DoF and is expansive [Stre 03]. A pseudo-triangle is a polygon drawn in the plane which has exactly three convex vertices. A vertex of a polygon is convex if the (internal) angle between the edges of the polygon at a vertex is strictly less than 180 degrees [Stre 03], see Figure 6-2. At this point Streinu makes use of the results of Günter Rote about expansive motions. Expansive motion means that no vertex-to-vertex distances decrease. A (pointed) pseudo triangulation without a convex hull edge is a 1DOF expansive mechanism which is proven in a theorem of Maxwell 1870’s.

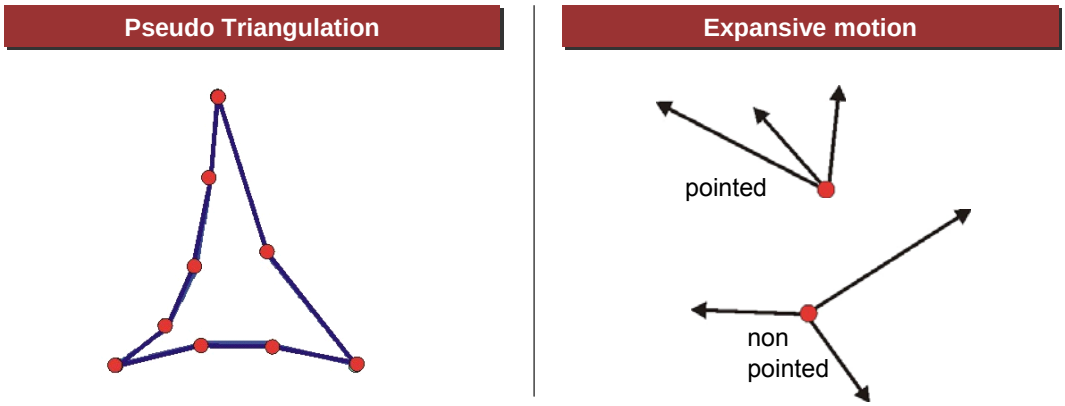


Figure 6-2: Triangulation, expansive motion

To start unfolding this bar-and-joint framework the pseudo triangulation-based mechanism will be moved along its unique trajectory in configuration space until two adjacent edges align. This means that the angle at a corner of a pseudo-triangle becomes zero, see Figure 6-3. At that point, a local alternation restores the pseudo triangulation.

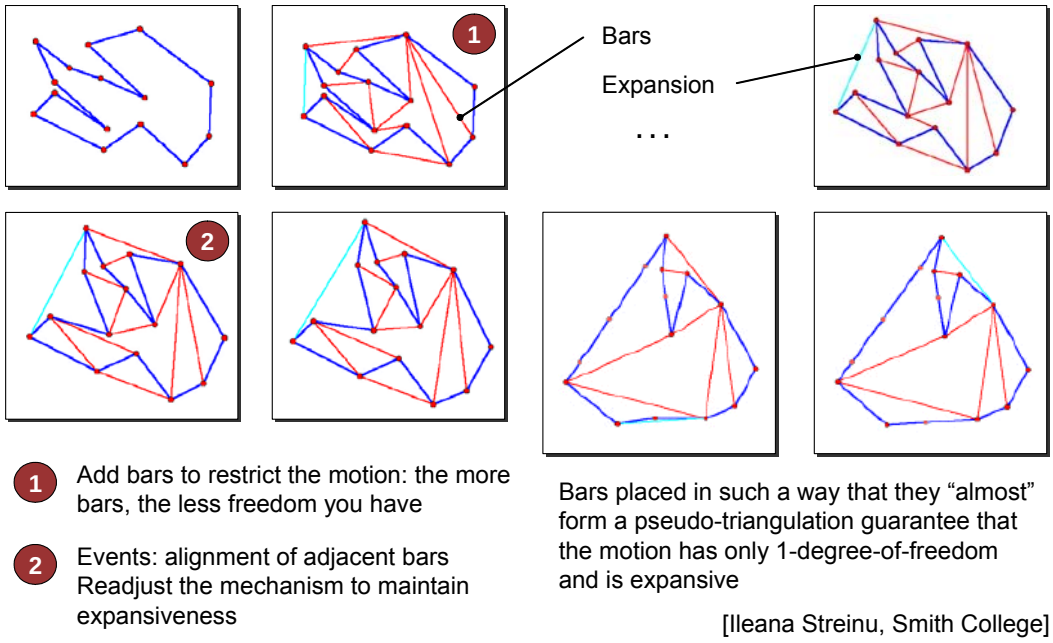


Figure 6-3: Unfolding on the base of the triangulation approach

These points are called “event points” and readjustments takes place. The motion continues for $O(n^2)$ steps until all the points are in convex position and the object is expanded or so called unfolded. This approach has great potential to verify the given 3D structure. If the bar-and-joint framework is expandable by avoiding collisions it is obvious that one can conclude that this structure is as well foldable from 2D back into its 3D initial shape.

With the use of these two fundamental concepts of folding and unfolding of polygonal linkages all applications which have been drawn up during the use case analysis are realizable. Given polygonal linkage configurations could be verified if they represent foldable structures and the following folding operation can be executed on the basis of the above presented kinematical description.

In addition, the general software architecture to verify and visualize the folding and unfolding operation on the basis of the above developed concepts will be pointed out. Hereby the implementation of the kinematical concept in terms of software architecture as well as its further realization is taking center stage.

7 Software Architecture of a General Solution

The two main concepts concerning the folding and unfolding of polygonal linkages are set. Consequently the theoretical consideration of the two previously analyzed use cases is completed. At this point we anticipate that the result of the above developed concepts will be realized in a comprehensive software package. The software should permit the easy, hands-on definition of foldable structures and their complete kinematic and dynamic analysis for design and verification in any given manufacturing challenge. To achieve this goal a preliminary functional specification is necessary. Hereto we will be geared to the results of chapter 4 concerning paper modeling to define the standard of a software package.

On the basis of these requirements the general software architecture will be developed. At this point we state clearly that the presented architecture as well as implementation is restricted to the folding of polygonal linkages. The pseudo triangulation approach will be shortly addressed in the general configuration of the software architecture.

Continued in chapter 8, a set of useful technologies will be presented in order to realize certain points of the architecture. In addition, the specification of the design will be presented in detail. Here the implementation of the data structure as well as the information flow between the different components is taking center stage. Concluding, the functionality of the newly developed software package will be verified in terms of an example scenario.

7.1 Requirements on Software Architecture

The analysis of the paper modeling and the adjacent folding process leads to classify the requirements into two category groups. The first subject area comprises the requirements to create a crease model whereas the second category specifies the requirements to initiate the folding process and the subsequently graphical visualization of the folding motion.

The specifications to model a user-defined crease pattern arise from the consideration of creased origami sheets of paper. An origami crease pattern is characterized by different types of crease lines. Their arrangement on a paper represents the placing of a fold whereas their type (dashed/dotted) reflects the orientation. To model this proceeding there is a data model required that enables to assign both, the position of

a crease line in reference to a “virtual” sheet of paper and the orientation (mountain or valley) of the fold. After the definition of the structure has been completed, the modification, add on or removal of already rendered creases represents a further specification. Consequently the complete generated crease structure needs to be stored in a consistant data model to ensure reusability.

Requirements of the Crease Model	Requirements concerning the Folding Functionality & Graphical Visualization
• Creating user-defined crease structures • Modify, add on or remove existing pattern • Consideration of origami terminology • Consistence data model to ensure reusability • Texture Mapping for photorealistic simulation	• Verification of the crease structure • Creation of a substituted mechanical system to apply the kinematical concept • Visualization of the complete transformation from 2D into 3D as well as to display individual motion segments upon request • 2D visualization of the crease synthesis • 3D visualization of the folding motion • Zoom in/out, look at all model views in 3D

Figure 7-1: Functional specification of the software

Concluding, a color or texture mapping of the virtual sheet of paper is desirable. In analogy to an origami paper sheet which is featured with its flipping colors of both sides, an individual and as well photorealistic reproduction of the folding process is accomplishable herewith.

After the crease model has been defined, a confirmation is required to ensure that the generated crease pattern represents a valid folding structure.

Next a graphical user interface is needed to adjust the three main folding properties to ensure the affiliated kinematical computation:

- Specification of one element that should be connected to the ground
- Choice of the axis of rotation that initiates the folding motion
- Specification of the co-domain of the twist angle

In addition, it is essential to link the data structure of the user defined crease design in such a way that a substituted mechanical system can be applied to guarantee the kinematical computation with use of the previous adjusted settings.

As a result the kinematical computation can be executed. Further on the software should dispose of the capability to visualize the complete transformation from a 2D

initial state into the 3D final shape. This requirement demands to be able to compute and visualize a variety of single folds over a specific twist axis in a row. Further on it is preferable to display individual motion segments available on request. The gratification of this requirement results in a folding simulation that is able to imitate a real “hand” fold. The skillfulness of professional origami designers to fold first about one axis, then about another and subsequently again about the first one to complete a fold could be retraced with this software in ease. Thereby the perspective of the visualization should be free of choice. In order to permit the creation of a crease pattern in analogy to an origami designer who looks perpendicularly at a sheet of paper during the creation of crease patterns, a 2D visualization of the crease synthesis is desirable. Accordingly a 3D visualization is demanded to simulate the folding motion. Furthermore the graphical visualization should enfold the functionality to zoom in and out as well as to modify the perspective of the model view even during the folding operation to achieve a maximum transparency of this proceeding.

7.2 Architecture Concept

Ongoing the above listed functional requirements will be implemented in modular constructed software architecture. The design of the architecture will be realized according to the classification of the system specification but makes as well allowances in its configuration for an implementation of the unfolding concept in the future.

Thus the overall software design can be subdivided into two main parts, the unfolding- as well as the folding of polygonal-linkages. Each part will be implemented in terms of a component based architecture which consists of a trisection. In the following we will focus on the implementation of the forward folding architecture and point out its configuration in detail. Hereby the tripartition comprises a module to ensure the generation of a crease structure as well as one transformation model to make sure that the crease configuration is available in a processable design for the inverse kinematics computation. Completing, a visualization model is used to display the motion of the folding operation on the basis of the kinematic results.

This modular architecture contributes to an easy expansion of the application in order to append further functionality, such as the incorporation of the dynamics or a self collision detection during the folding operation. In the following we will illustrate the design of each component in detail and take up with the crease component.

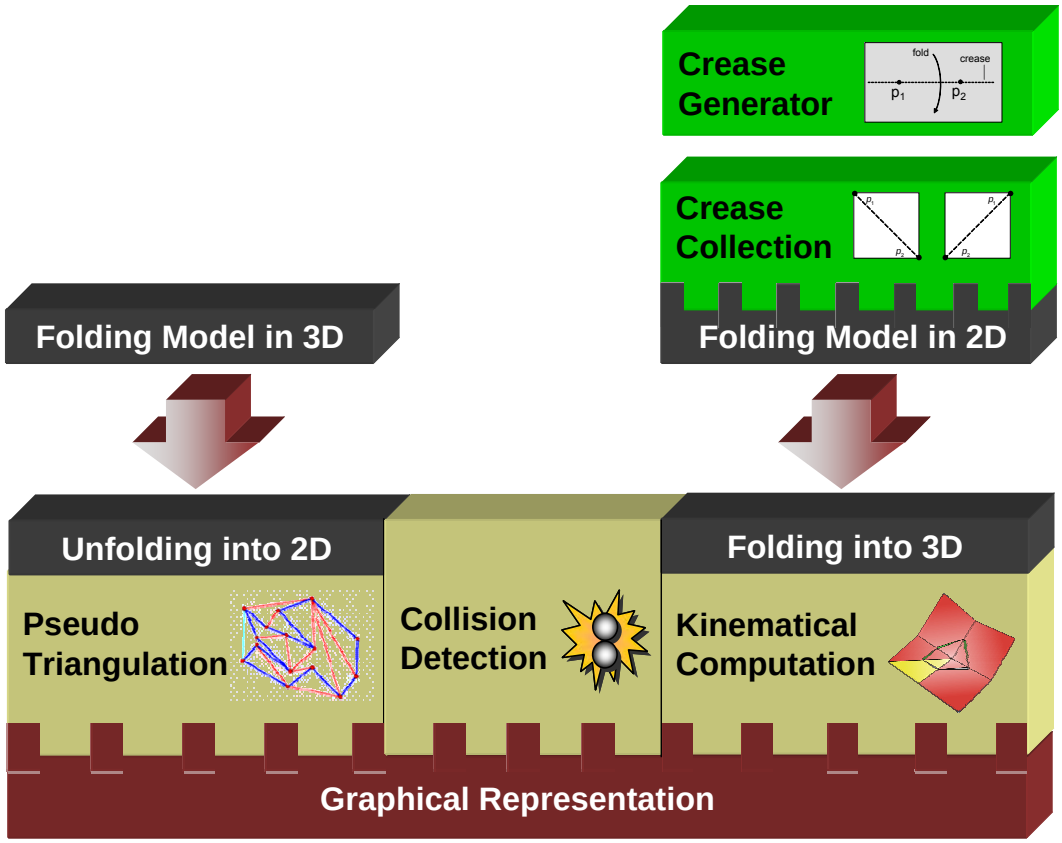


Figure 7-2: General Software Architecture

7.2.1 Crease Component

The creation of a crease structure is subdivided into two components, the “Crease Generator” (CG) and the “Crease Collection” (CC). The crease generator is responsible for the structural design of a crease and delivers as a result a collection of adaptive crease patterns which are stored in terms of an object in the crease collection component. Herewith we pursue the idea to bundle the manifoldness of Huzita’s axioms in one component in order to permit the creation of user-defined crease structures. The advantage of this “string together technique” is its reusability. Complex crease structures can be defined once and reused in terms of a collection object by applying this pattern to a section of the virtual sheet of paper. Herewith the generation of a user-defined crease pattern becomes possible with minimum effort.

The use of axiom number one, for example, allows generating a unique fold that passes through two user-defined points p1 and p2. Depending on the position of these two points a user can determine different fold structures. Figure 7-4 illustrates the parametric formulation of four possible crease designs which arise from the crease generator component by using axiom number one.

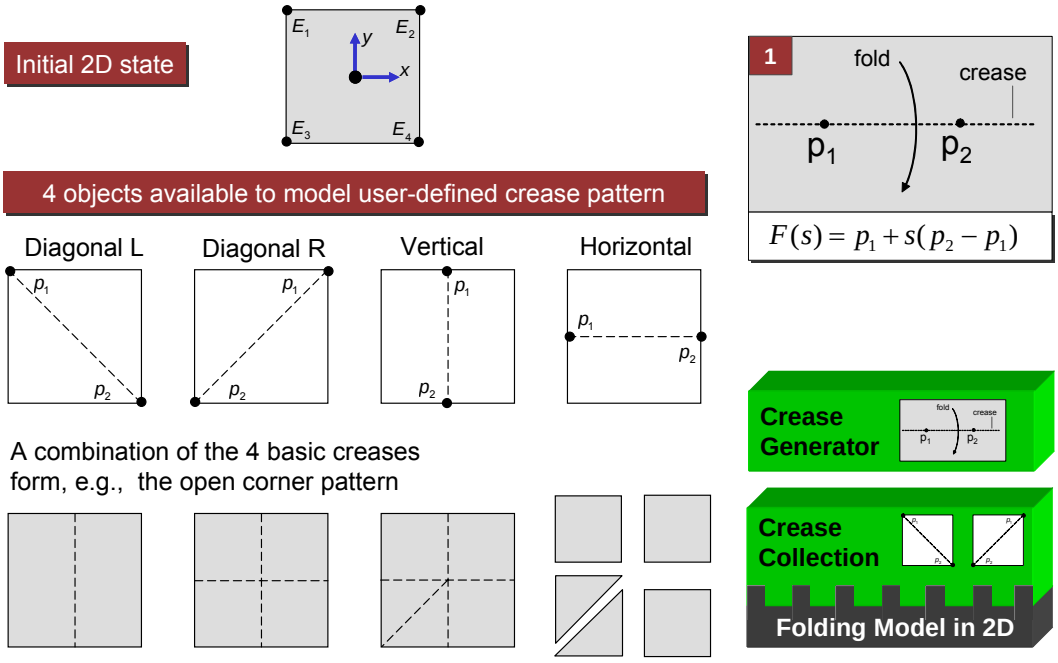


Figure 7-3: Crease Generator Architecture in detail

This parametric information will be stored in terms of an object in the crease collection component. Assuming the user would like to create the crease pattern of an open corner, he uses these objects in order to string together the diagonal L -, the diagonal R -, the vertical and a horizontal object in the right order to receive his desired crease pattern.

$$F_1(s) = \begin{pmatrix}-1 & 1\end{pmatrix}^T + s\left(\begin{pmatrix}1 & -1\end{pmatrix}^T - \begin{pmatrix}-1 & 1\end{pmatrix}^T\right) \quad \text{Diagonal Left crease}$$

$$F_2(s) = \begin{pmatrix}1 & 1\end{pmatrix}^T + s\left(\begin{pmatrix}-1 & -1\end{pmatrix}^T - \begin{pmatrix}1 & 1\end{pmatrix}^T\right) \quad \text{Diagonal Right crease}$$

$$F_3(s) = \begin{pmatrix}0 & 1\end{pmatrix}^T + s\left(\begin{pmatrix}0 & -1\end{pmatrix}^T - \begin{pmatrix}0 & 1\end{pmatrix}^T\right) \quad \text{Horizontal crease}$$

$$F_4(s) = \begin{pmatrix}-1 & 0\end{pmatrix}^T + s\left(\begin{pmatrix}1 & 0\end{pmatrix}^T - \begin{pmatrix}-1 & 0\end{pmatrix}^T\right) \quad \text{Vertical crease}$$

Figure 7-4: Four folds created with the CG and stored in the CC

After one crease object has been applied to a virtual sheet of paper its geometrical shape is split into two new elements (obj1 and obj2), see Figure 7-5. Depending on the parametric formulation we differentiate between three basic geometries coming into existing, after a crease pattern has been applied. We distinguish between a rectangle, quad or any arbitrary polygon. This means, the actual function of a crease pattern is similar to a perforation. Therefore we keep tracking two strategies; at first we extract the geometrical shape of each component which comes into existing and second we are interested in the corresponding rotational axis of each element. In

order to characterize each component we use five major elements which get stored in the so called Geometry Database (DB). First of all, an identification code (ID) is used to ensure a definite identification, second the geometrical type will be stored as well as the respective points to draw on this geometrical shape. Further on the axis of rotation in terms of the parametric formulation of the crease line will be saved. In order to complete the description a specific color or texture could be applied to enhance the illustration during the visualization.

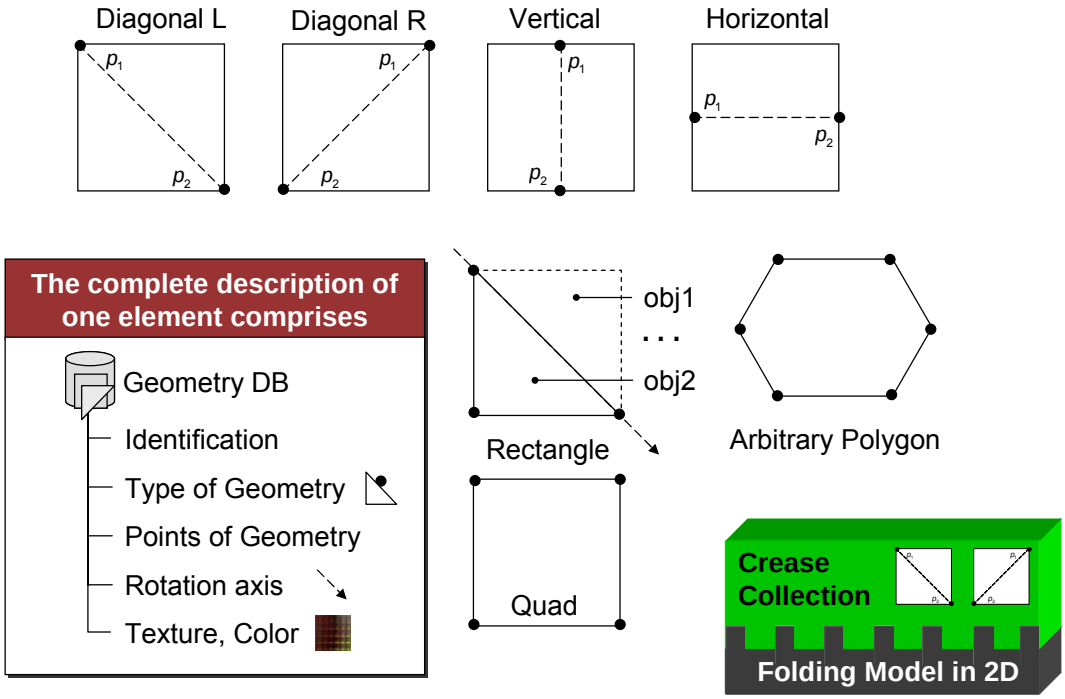


Figure 7-5: Applying a crease pattern

7.2.2 Transformation Component

Next a transformation component is needed to realize the idea mentioned in chapter 5, to uncouple the “paper” aspect from the folding operation by introducing a substituted mechanical system. Hereto the transformation component converts the above presented geometrical description of a user-defined crease pattern into a compatible format for a substituted mechanical system. This conversion is based on the idea to replace each crease element by a bone vector. The bone vector can be specified by detecting these points which are in common to the current element and its two neighboring elements, so the coordinates of the vector result to:

$$\frac{P_{5,1}-P_{5,2}}{2}=B_{1,x} \text{ and } \frac{P_{2,1}-P_{2,2}}{2}=B_{1,y} \qquad (1.6)$$

This “uncouple” concept leads to the fact that the complete crease structure can be replaced by a substituted mechanical system or so called bone structure. The degree of complexity will be decreased because we do not need to describe the motion of a complex geometry with an unknown number of points. Each arbitrary element will be replaced by a bone consisting of only two points $B_{i,x}$ and $B_{i,y}$, see Figure 7-6.

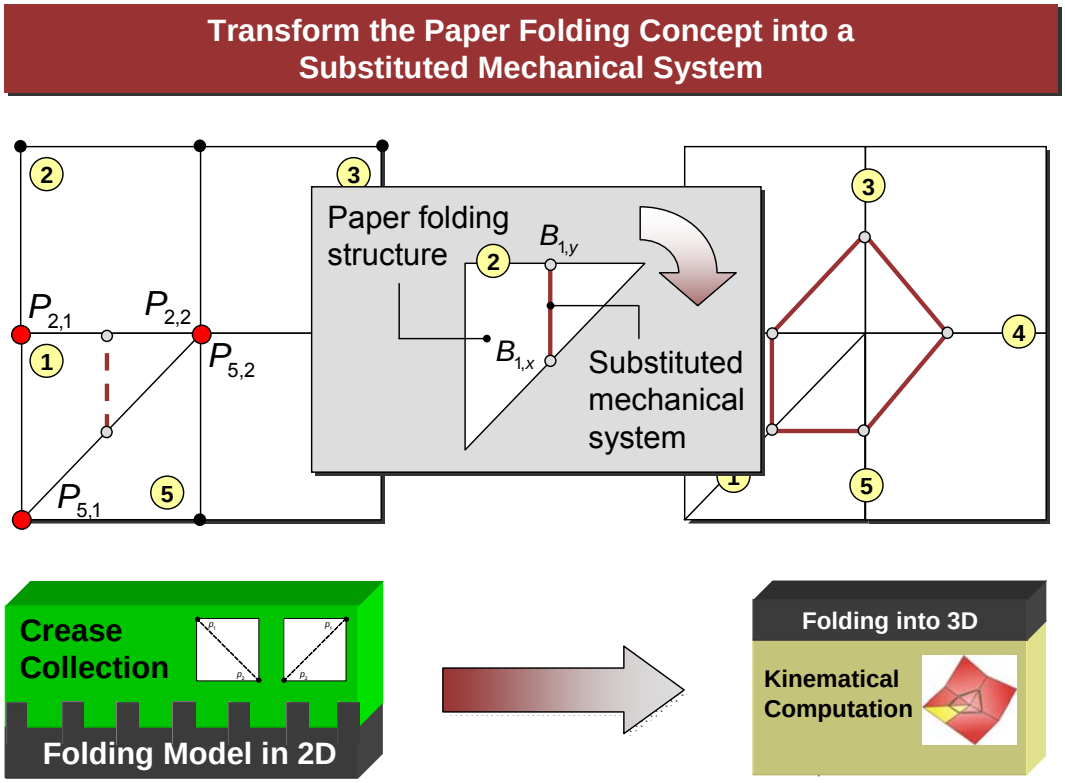


Figure 7-6: Transformation concept

The geometrical information of the bone structure will be saved in the corresponding Bone Structure Database (DB). In order to proceed in the realization of the kinematical concept, we have to provide the possibility to split the computed bone structure into a forward kinematic and inverse kinematic part. Hereto there are three additional constrained conditions necessary to transfer the developed mechanism into a first degree of freedom system. The software needs an arrangement to receive user input concerning the information which element will be fixed to the ground, which axis of rotation will be locked as well as information about the dimension of the twist range; hereby each participating bone has to be expressed accordingly to its reference system (frame A, frame B). Thereupon the computation of the corresponding segments can be executed and the result is stored in the Kinematic Frame Database (DB) consisting of five fields, see Figure 7-7.

.

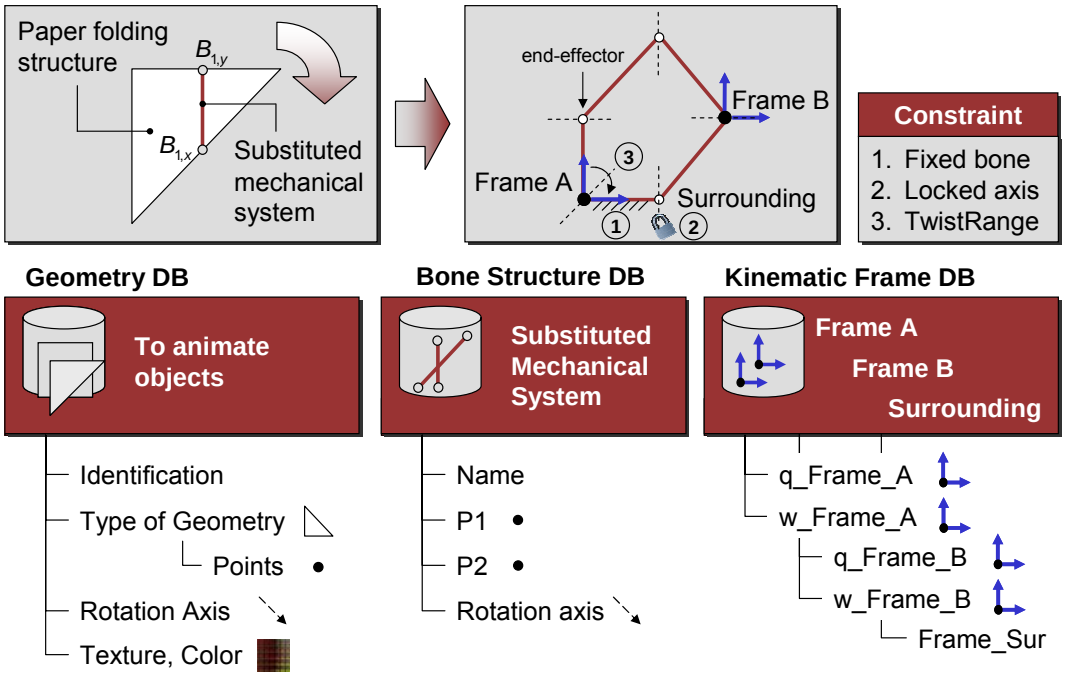


Figure 7-7: Data architecture in detail

Four fields are designed to save the necessary parameters (q_i, ω_i) for the matrix exponential notation for both the forward and inverse kinematic chain. The remaining bones which do not participate in this current folding are stored in the data field “surrounding”.

7.2.3 Kinematic Component

After the mechanism has been broken down into frame A and frame B the actual kinematical computation can be accomplished. Hereto a mathematic engine is used to compute the necessary calculations. For this reason the kinematic component sends the dedicated geometrical information of frame A and - B to a mathematic engine.

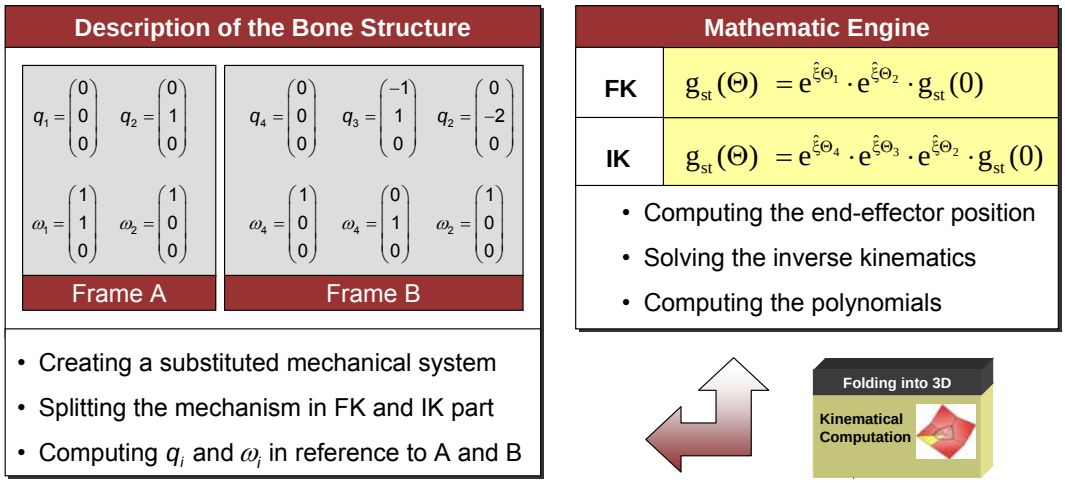


Figure 7-8: Kinematical Concept

This computes the end-effector position of a given twist range, solves the inverse kinematics and as a last step calculates the corresponding coefficients of a curve fitting through the evaluated supporting points. Hereupon the kinematic component receives a return value in terms of the polynomial coefficients and builds up the transfer functions of the twist angle and its corresponding constrained angels. With the use of this system of equations the visualization of the folding operation can be executed for any arbitrary angle inside the twist range.

7.2.4 Visualization Component

With the use of these transfer functions the visualization of *one* folding operation can be realized. Depending on the number of elements the inverse kinematic chain is consisting of, the same number of transfer functions becomes available to compute the corresponding constraint(ed)? angles for a given twist angle. In a next step a graphic engine will be connected to the above mentioned databases in order to visualize the geometric information. This linkage between the database configuration and the graphic engines ensures a consistent image of the folding process at any time. With the use of a scroll bar that generates a set of twist angles inside the precompiled twist range, an animation of the folding operation becomes available. But the challenge is to visualize the complete transformation from a 2D initial state into a 3D final shape.

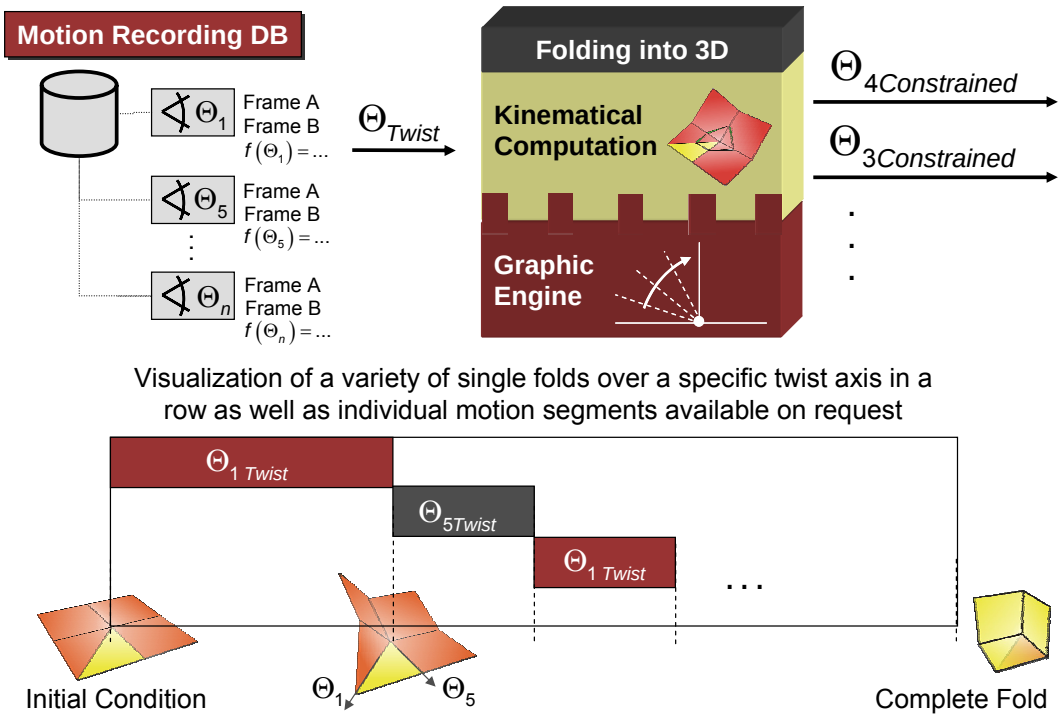


Figure 7-9: Motion recording concept

Therefore the software requires the feature to visualize a variety of single folds over a specific twist axis in a row as well as to display individual motion segments available on request. This requirement will be met by using a Motion Recording component. The Motion Recording database stores the coordinates of frame A, frame B and the Surrounding in space as well as its corresponding transfer functions after a single fold has been completed. On the basis of this information a new fold about another twist axis can be planed, computed and executed. Further on this strategy facilitates the restoration of the initial condition of each fold. Herewith the jump into any arbitrary position inside the folding sequence becomes available and animates a complete fold beginning with the 2D initial condition until the final 3D shape is reached. The motion recording component represents an essential part of the architecture and ensures maximum transparency of the folding operation.

Up to this point all requirements which have been enumerated in the functional specification are implemented in general software architecture. In the next chapter the realization of this architecture will be discussed.

8 Implementation of the Architecture

After the general architecture has been established the implementation of the software follows. In the first place, some technologies will be presented on which the realization of the architecture is based. Ongoing the integration of specific components into the development environment will be pointed out and the Graphic User Interface (GUI) will be illustrated. Then the functionality of each component will be broken down to its bottom layer and its interrelationship will be visualized in a dataflow diagram. Further on the realization of each specific component will be explained in detail while focusing on the implementation of the GUI as well considering the programming aspects.

8.1 Technologies to Implement the Architecture

There are three basic technologies required to start implementing the architecture. A development environment has to be chosen first, second a dedicated mathematic engine which ensures the data exchange in two directions, from the host application to the mathematic engine and vice versa. Further on a graphic engine is needed to enable the graphical representation. An analysis of diverse distributable software tools reaches in the following result.

The Implementation of the architecture is based on the Visual Basic development environment. This programming language in combination with OpenGL as embedded component is highly suitable for the creation of interactive 2D and 3D graphical applications. Since its introduction in 1992, OpenGL has become the industry's most widely used and supported 2D and 3D graphics application programming interface (API) [Open 03]. The extensive mathematical computation of the kinematics will be implemented by making use of the mathematic software tool, Matrix Laboratory (MATLAB). Here the choice for MATLAB is due to its modular architecture which is highly adapted for the implementation into our configuration. The use of the MATLAB Mathematical Function Library which represents a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, and fast Fourier transforms can be utilized to implement the kinematical computation [Math 03].

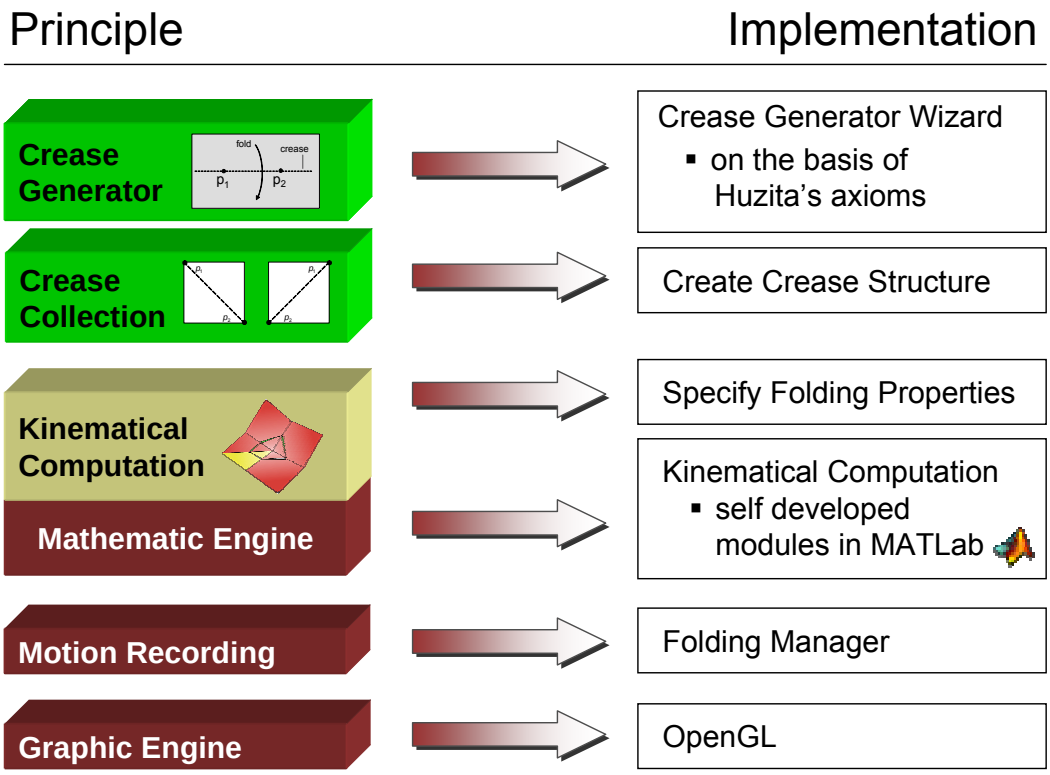


Figure 8-1: Technologies to implement the architecture

In combination with the MATLAB Application Program Interface (API) which includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files the data exchange is ensured. These three basic technologies build up the framework of the realization.

On the basis of these basic conditions the implementation of the above developed architecture will be carried out. The individual components of the architecture will be realized in corresponding Visual Basic modules, see Figure 8-1. The theoretical consideration in chapter 6 which results in a trisection to describe the folding process will be extended by further intermediate steps to enable an easy to use GUI.

Hereto the realization of the kinematical computation, for example, will be split into two parts, one to specify the folding properties and a second which is responsible for the computation, see Figure 8-1. The overall realization is based on the strategy to disassociate basic functionality from data management problems as well as from graphical functionality. This configuration has the advantage of being able to replace only certain components of the software if functional requirements are changing. Ongoing in the following chapter the realization on the basis of the above presented technologies will be pointed out in detail.

9 Realization of the Software Architecture

The required technologies to realize the architecture are set. In a further step of development the software architecture will be transferred into a suitable configuration which can be implemented in Visual Basic and is called OrigamiKinematics^{Plus}. Here the main focus of the implementation is directed to design a GUI which enables the creation of an arbitrary crease pattern and to fold the model along the creases into the desired final 3D shape. The handling of the software which means the consecutive steps to initiate the folding should be executable similar to a real paper fold, see chapter 4.1 (modeling a paper fold). All intermediate steps like fixing one element to the ground which is natural for an origami designer but necessary to enable a unique computation, will be inquired by the GUI. Hereto the GUI is connected with up to four components including the actual functionality to model and visualize a complete transformation from 2D into 3D, see Figure 9-1. The first component provides the possibility to create a crease pattern on the basis of a predefined crease collection. At this point we refer that we disclaim realizing the crease generator concept in a first step of development. We assume a standard crease collection which is based on Huzita's axiom number one, as mentioned in chapter 7.2.1, in order to generate elementary crease structures.

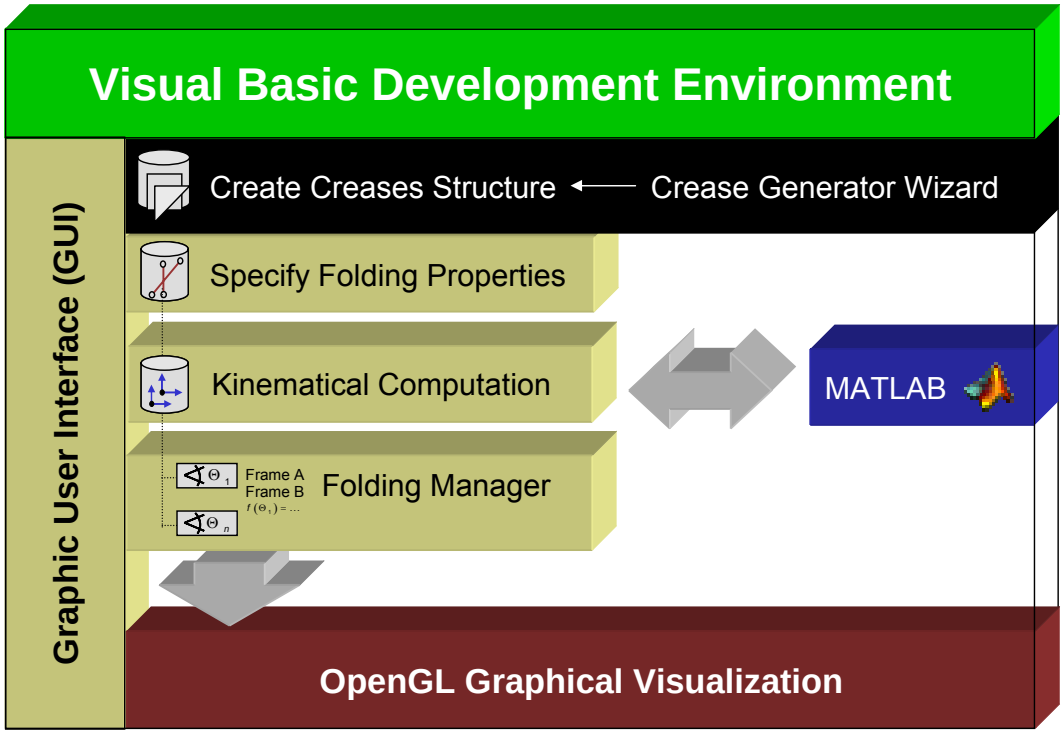


Figure 9-1: Realization of OrigamiKinematics^{Plus}

A second component “Specify Folding Properties” queries the element that should be connected to the ground, the axis of rotation that initiates the folding motion, and the co-domain of the twist angle to ensure a unique calculation. Next the kinematic component takes up its work and dispatches the frame parameters to an outsourced MATLAB module overtaking the actual computation. The “Folding Manager” component records the specific computations and enables to restore any arbitrary initial state of a folding operation. At last an embedded OpenGL module visualizes, optional in 2D or 3D, the folding operation. In order to comprehend the configuration and implementation of each component in detail, we will introduce first the design of the graphic user interface to get an impression of the software layout. Next the interrelationship of the specific components are explored in a data flow diagram and in addition the implementation of each component will be explained in detail.

9.1 Graphic User Interface

The design of the graphic user interface is oriented to guide the user step by step from defining the crease pattern over adjusting the necessary folding properties up to retracing the folding simulation. Hereto the layout of the GUI is classified into five category groups, see Figure 9-2.

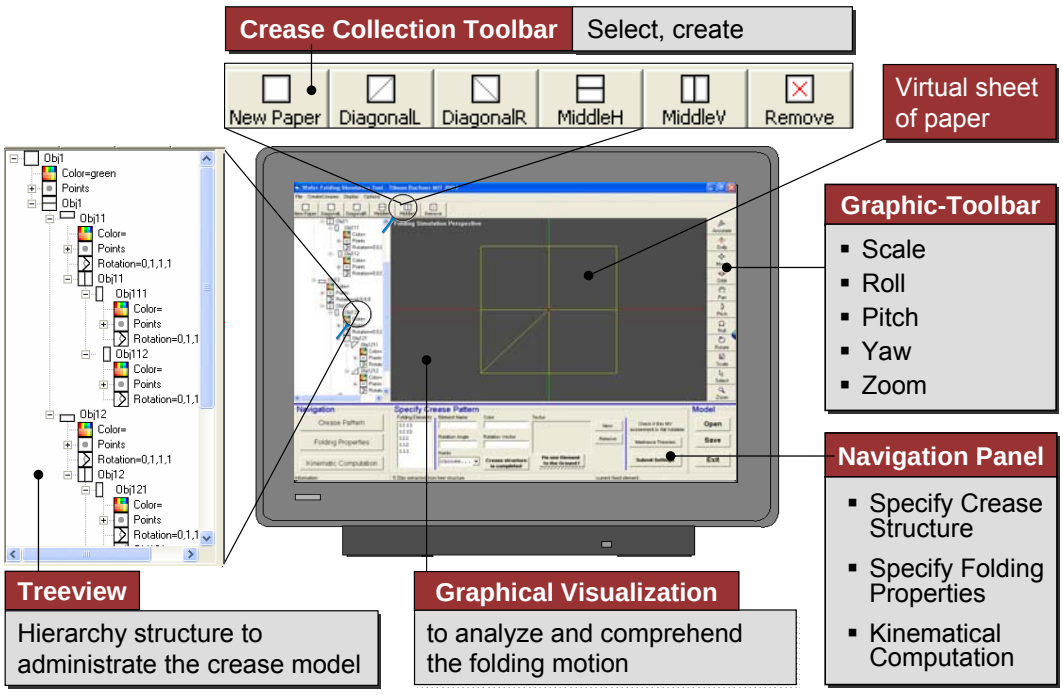


Figure 9-2: Graphic User Interfac in detail

A toolbar arranged on the top of the screen makes it possible to elect from a predefined collection one specific crease pattern which can be applied to the current virtual sheet of paper. A hierarchical tree view on the left permits the administration of the crease pattern. Creases may be removed or existing patterns may be expanded by applying further predefined patterns of the toolbar. In the bottom margin the navigation panel is located in order to control the complete functionality of the software at a glance. The navigation panel comprises three frames, one to complete the creation of a crease pattern, one to specify the folding properties and at last one panel to initiate the kinematical computation and to control the folding simulation, see Figure 9-3. The visualization of the folding operation can be retraced in the graphic display in the middle of the screen. A toolbar on the right makes it possible to zoom in or to rotate the model in an arbitrary direction by choosing one of the corresponding buttons roll, pitch, yaw, etc. In the menu bar the user can save the created or open existing models. The Folding Manager in the “Option” menu can be used to control the complete transformation from 2D into 3D.

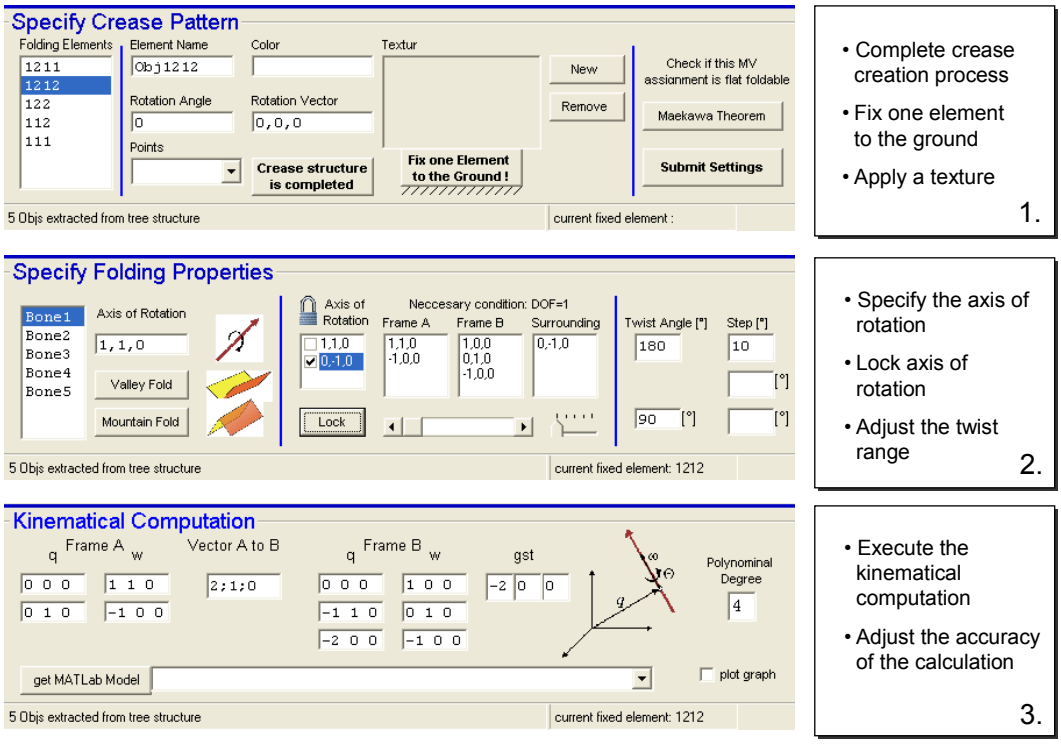


Figure 9-3: Navigation Panel in detail

The exact handling of the software can be retraced in the example scenario in connection to this chapter. In a further step the dataflow behind this front-end will be explained in detail by focusing on the functionality of the visualization component and the way the data management is organized.

9.2 Dataflow at Run-Time

The dataflow diagram in Figure 9-4 reflects the component based architecture. The dataflow can be categorized into data management, functionality, and visualization aspects. Here the focus of the architecture represents the embedded OpenGL component which is passed trough periodically to ensure a consistent image of the data management. The associated “Draw” method serves to read out the different data storages according to the current flag (e.g. DisplayFrameABsur=True) to provide the geometrical information for the OpenGL graphic routine.

The other modules are responsible for the functionality and insert the corresponding information of the use case into the right data base. Here the “Extract Objects” as well as the “Complete Crease Model” module provide the necessary geometrical information, in order to describe the shape of the folding model. These data represent the base of operation for the “Fix one Element to the Ground” component which transfers the geometrical information into a suitable format to generate a bone structure. If the operation is completed the current flag switches to DisplayBoneStrucutre=True and the OpenGL component visualizes the bone structure. Next the user can specify the folding properties and as a result the “Lock Axis of Rotation, Split Frames” module generates the corresponding information for the additional kinematic computation into the Kinematic Frame Database.

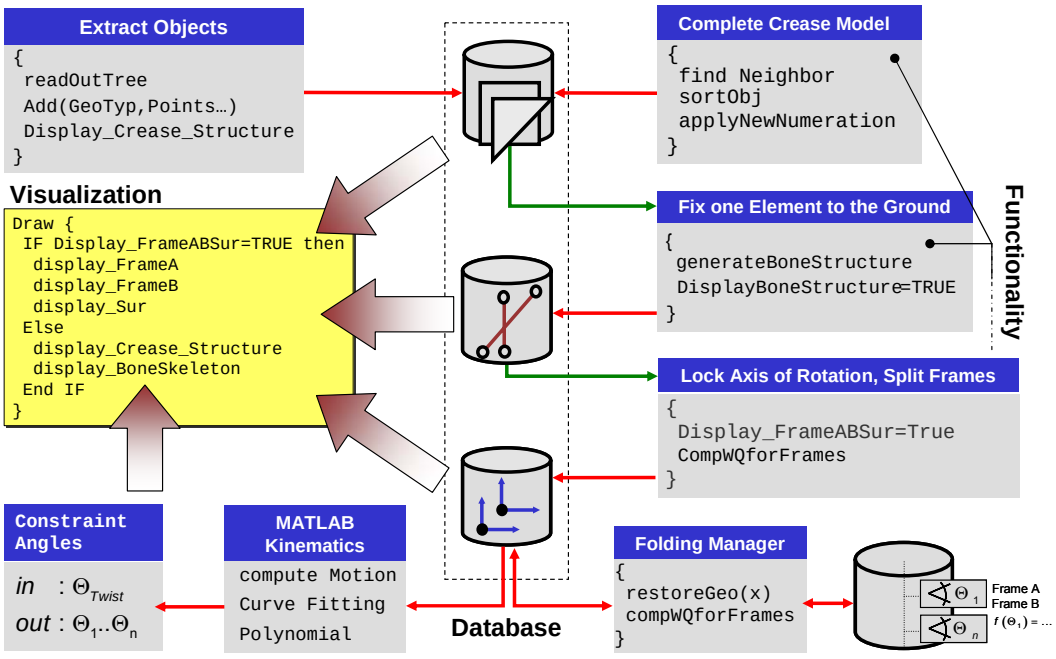


Figure 9-4: Data flow in detail

The MATLAB kinematic component accesses the information, executes the calculation and the visualization of the folding motion starts in combination with the input of the twist angel. The folding manager observes the computed frame parameter and stores each frame configuration in the Motion Recording Database which makes it possible to restore any arbitrary fold. In the following each component that participates in the functionality will be presented in detail.

9.3 Specification of the Crease Pattern

In the first place the user is able to choose any arbitrary crease pattern from the predefined crease collection toolbar and to apply the associated mathematical description to any tree entry. The hierarchical tree structure represents a consistent image of the current crease model. Here the user obtains the possibility to realize the complete crease structure at a glance. He is able to modify existing creases or to apply new ones by selecting a parent object in the tree structure and choosing the desired crease pattern in the toolbar.

Extract Objects

The tree structure follows a descending order. The caption of each object consists of the name and the degree of descending order (e.g. Obj11). If the user applies a crease pattern to Obj12 for example, the software subdivides this object into

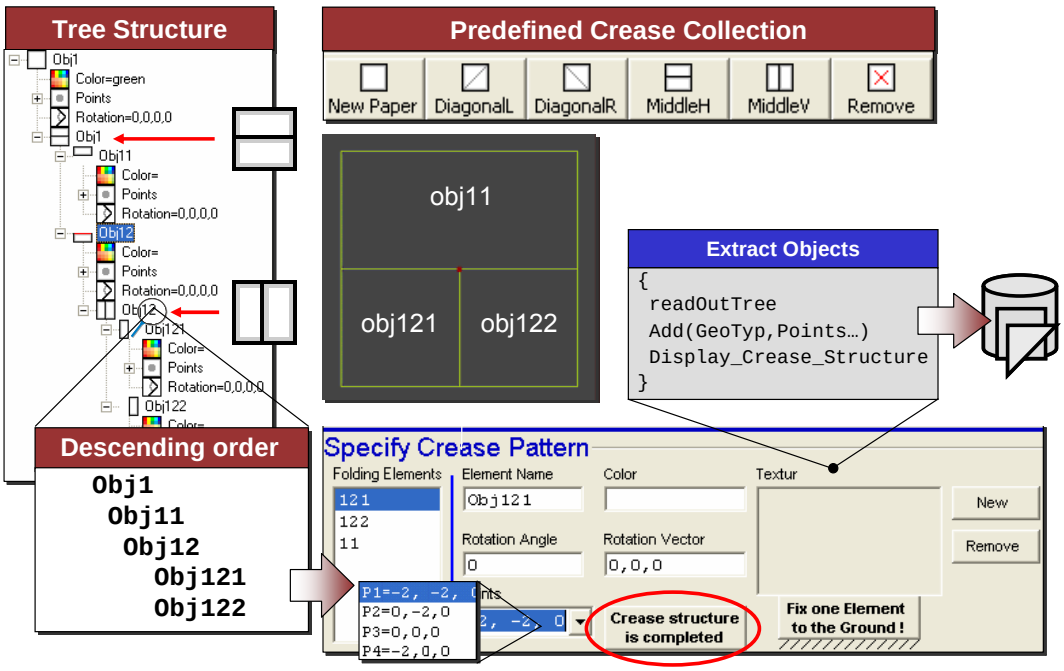


Figure 9-5: Create crease structure

two new objects which receive the name Obj121 and Obj122 accordingly to their parent. This strategy can be retraced in the complete tree structure. But targeting to visualize the tree structure, we are not interested in all of these objects only in those with the highest descending order, because these objects represent the elements which arise from the last allocation of a crease. The module “Extract” skips this part automatically after a new crease has been applied and updates the Geometry database. The user only recognizes new entries in the “Folding Elements” listing and the visualization component displays the content of the database, see Figure 9-5.

Complete Crease Structure

After the crease model has been completed, the user has to confirm this operation by selecting the button? “Crease Structure is completed”. Hereupon the module “Complete Crease Model” will be executed. This operation is necessary because the composition of the crease structure can be done in any user-defined sequence.

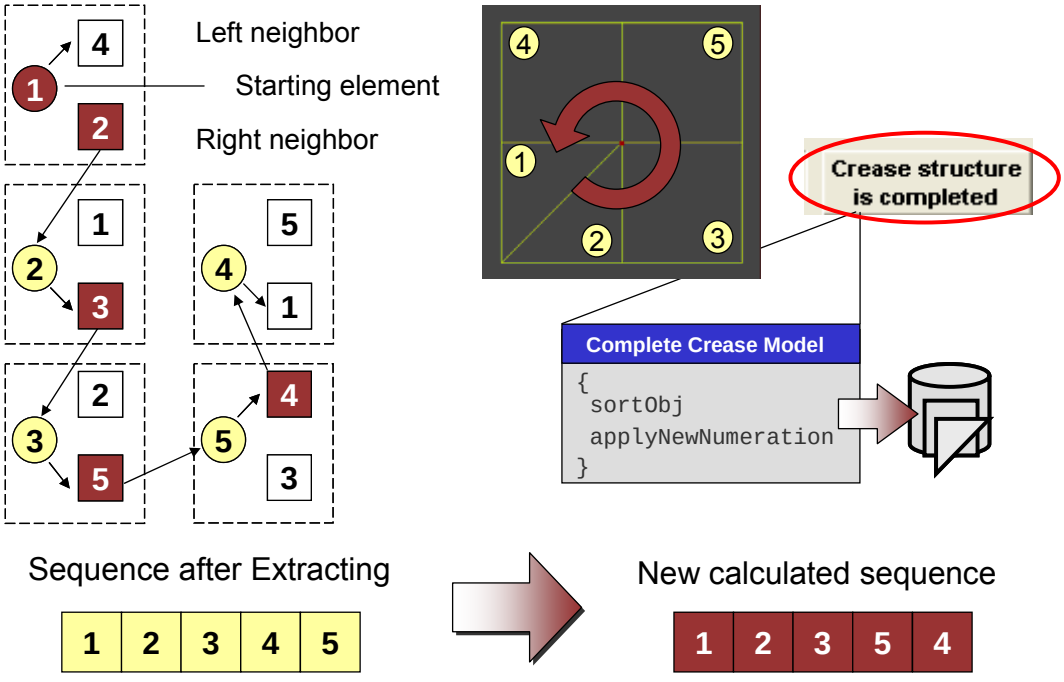


Figure 9-6: Compute consecutive sequence

According to this the extracted objects are presented in a non-systematic sequence but there is a need for a consecutive sequence to enable an adequate handling on going. Hereto the function “sort Obj” starts evaluating a consecutive sequence. The idea behind this method is to start with the right/left neighbor of any of the extracted elements. Then one iterates through all of these elements by looking for the corre-

sponding element which matches the right/left neighbor element. If this element is found one starts again this proceeding now looking for a new right/left neighbor element. This process is completed if the initial element is equal to the right/left neighbor of one of the scanned elements. Hereupon a consecutive sequence is detected and a new numeration can be applied to the Geometry database, see Figure 9-6.

Next the first constrained condition will be queried. The user has to specify the element which shall be fixed to the ground. Hereupon the module “Fix one Element to the Ground” starts searching for the neighbors of the selected element and computes the corresponding axis of rotation as well as generates the bone structure on the basis of the considerations in chapter 7.2.2. At this point the precompiled consecutive sequence advantages the operation to find the forerunner and back runner of a specific element. The result is stored in the Bone Structure database and will be displayed by a red line composition on the crease structure, see Figure 9-7. The corresponding axes of rotation will be appended to the respective list object in the “Specific Folding Property” frame. Herewith the crease pattern is completely documented and the specification of the folding operation can be followed.

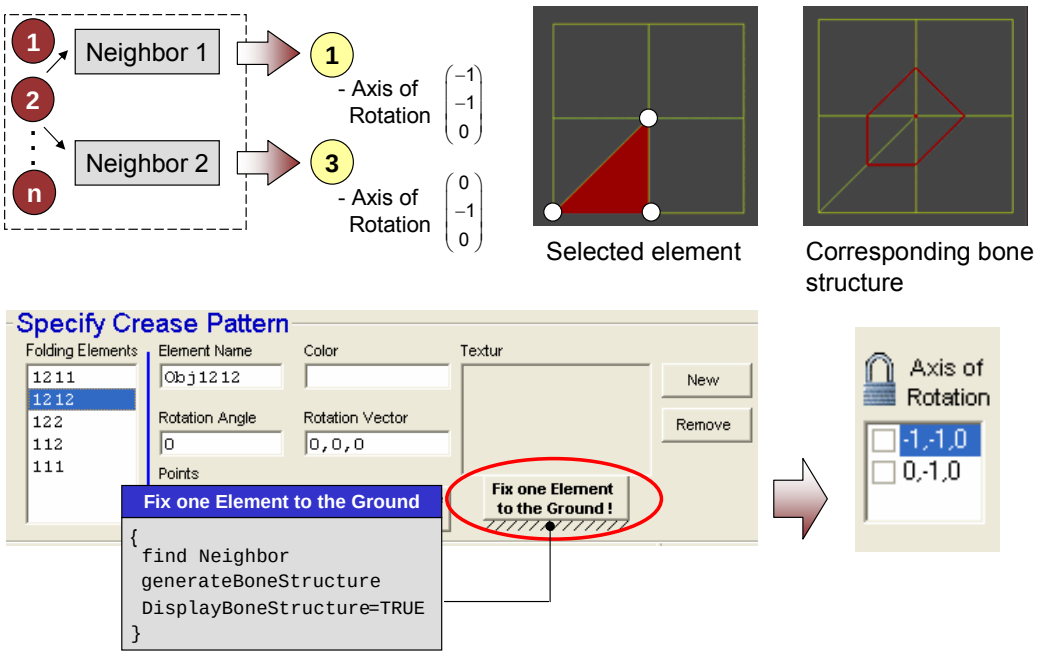


Figure 9-7: Specify crease pattern

9.4 Specification of the Folding Properties

First of all the user has to specify the orientation of the individual axis of rotation. According to the right hand rule the user can change the orientation by using the “Valley Fold” or “Mountain Fold” buttons. Further on he has to specify the twist range of the kinematic computation in degrees. The declaration of this step is important for the accuracy of the calculation. After all orientations are determined and the folding range is defined, one axis of rotation needs to be locked to fulfill the essential condition to reduce the mechanism’s degree of freedom to one. Hereupon the software starts reassigning the bone elements according to their participation (forward or inverse kinematic) into frame A, frame B and Surrounding, see Figure 9-8.

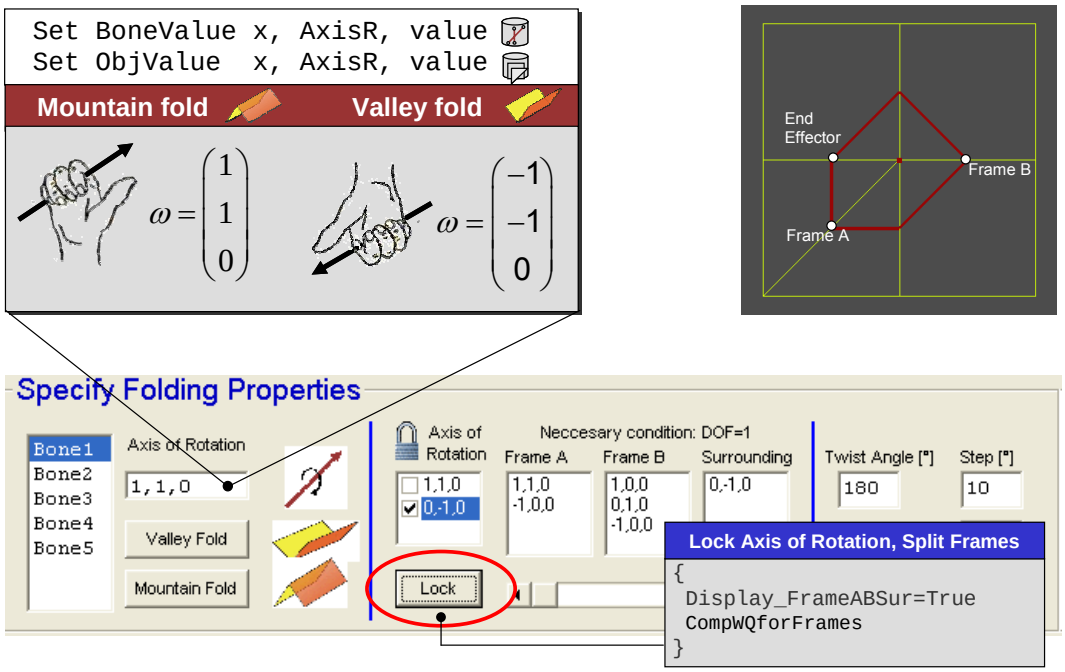


Figure 9-8: Specify Folding Properties

The fragmentation obeys the following rules. One starts with the axis of rotation which is being locked. This element does not take part in the current folding operation and belongs consequently to the surrounding group. According to this, the twist axis as well as its neighboring axis represents the two elements of frame A. Hereby the selection of the neighboring axis demands compensation with the current locked axis; according to this, one obtains the left or the right axis as the corresponding neighboring axis. Now the elements of frame B arise from counting up or down from the current surrounding element until the last element of frame A is reached, see Figure 9-9

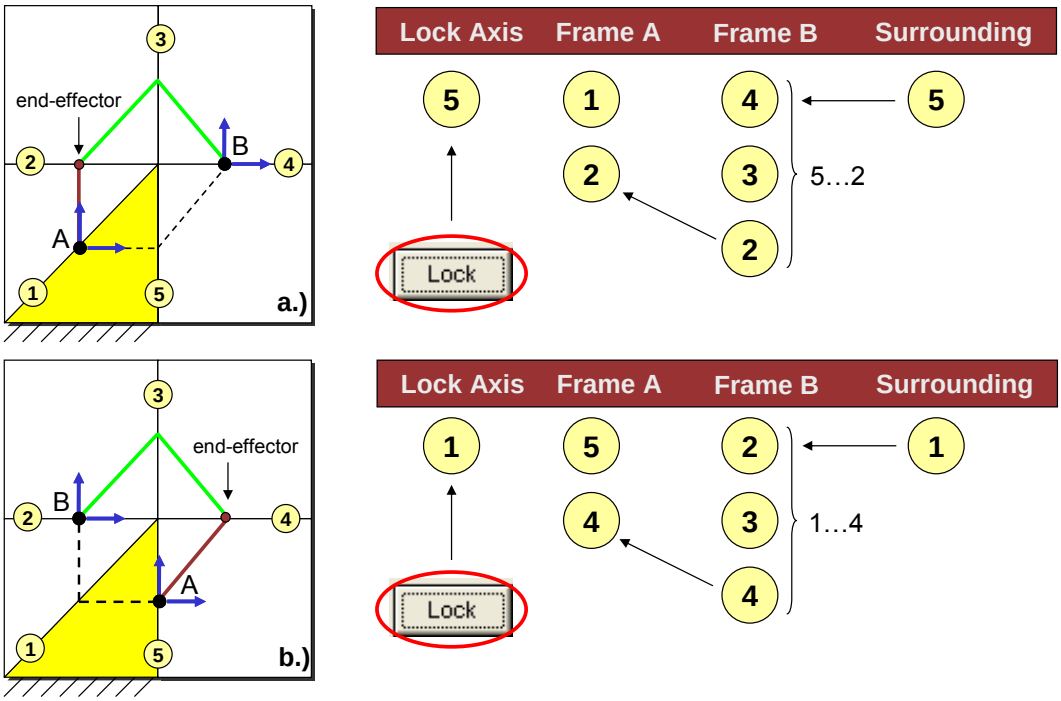


Figure 9-9: Segmentation of the axis of rotation

Counting up or down depends on the location of the locked axis of rotation (left or the right axis of the fixed element), compare Figure 9-9a and Figure 9-9b. After the segmentation has been completed, the referencing of the initial individual bone structure segments needs to be changed from the vertex to the previous identified frames.

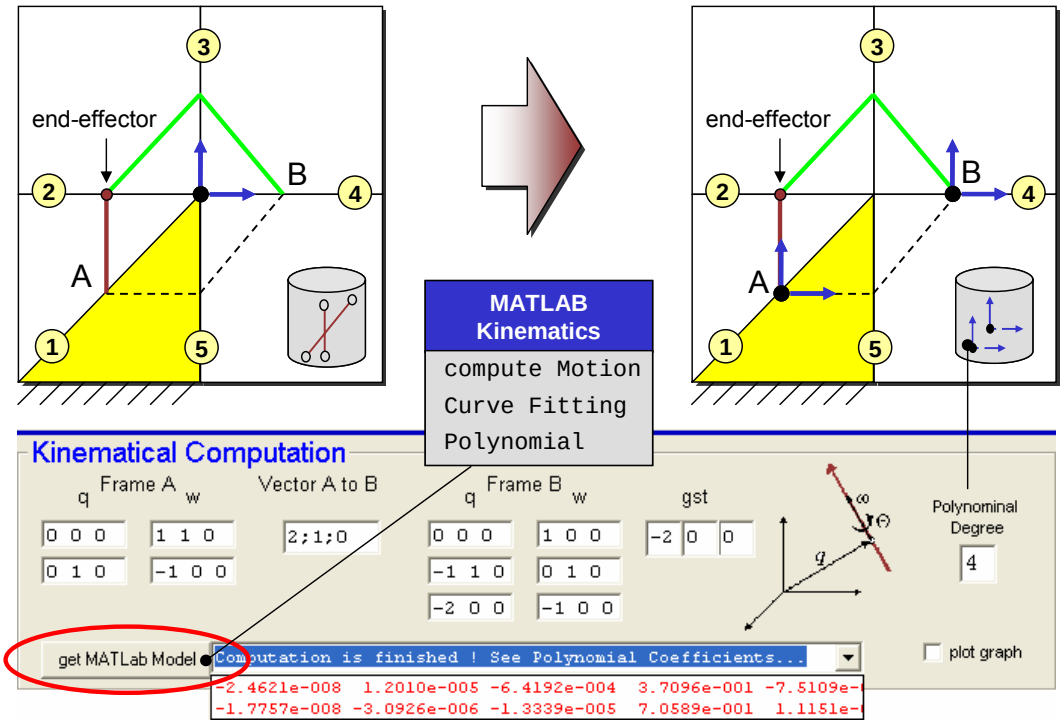


Figure 9-10: Transformation of the reference system

The result is stored in the frame database and represents the base of operation for the kinematical computation, see Figure 9-10. But before the computation can be executed, the module “compute_WQforFrames” determines the vector $\vec{v}$ pointing from frame A to frame B, the corresponding parameters q_i and ω_i and $g_{st}(\Theta = 0)$ to gather all necessary information for the following screw calculus computation.

9.5 Realization of the Kinematic Computation

As mentioned before the kinematic computation will be executed by an outsourced module developed in MATLAB. Before we present the dataflow of this calculation, we will introduce two possibilities to establish a data exchange with MATLAB.
One differentiates the possibility of communicating via a MATLAB COM Object or with the use of the MATLAB Automation server. Each alternative has its advantages and disadvantages. But at first we will draw the outline of these two technologies.

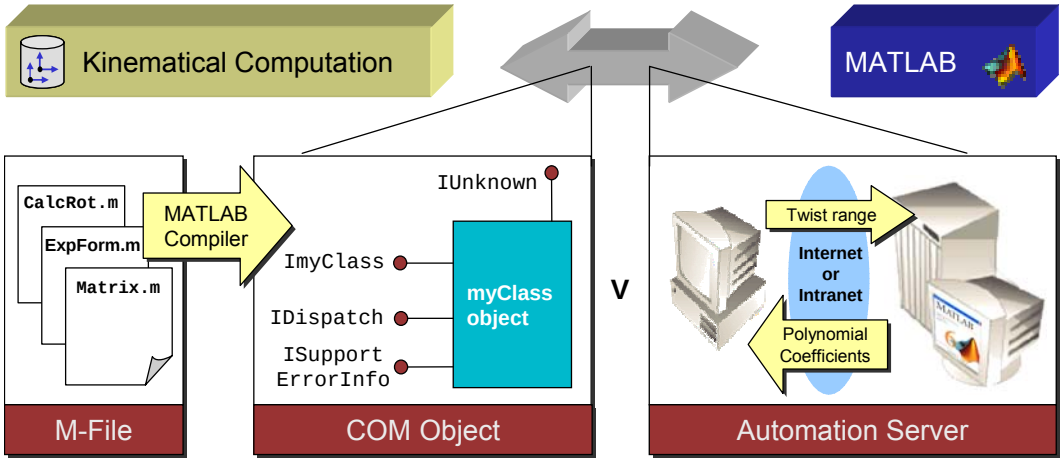


Figure 9-11: Two communication principles

The MATLAB COM Builder makes it possible to integrate compiled MATLAB models into Visual Basic, C++, or any other language that supports COM. COM is an acronym for Component Object Model, Microsoft's binary standard for object interoperability. COM is the widely accepted standard for integration of external functionality into Microsoft Office applications, such as Excel [Math 03]. The process of creating a MATLAB COM Builder component is completely automatic. The user supplies a list

of M-files to process and a few additional pieces of information, i.e., the component name, the class names, and the version number. The build process that follows involves code generation, compiling, linking, and registration of the finished component. The library can be linked into any Visual Basic project and the resulting call syntax from Visual Basic is systematically mapped to the syntax of the original MATLAB function.

The MATLAB Automation Sever on the other hand makes use of the Microsoft Windows COM Automation server capabilities. Automation is a COM protocol that allows one application or component (the controller) to control another application or component (the server). Thus, MATLAB can be launched and controlled by any Windows program that can be an Automation Controller. Using Automation, you can execute MATLAB commands, and get and put information from and to the MATLAB workspace [Math 03]. The following Visual Basic code fragment invokes the MATLAB Automation Execute method.

```
Dim MatLab As Object
Dim Result As String
Set MatLab = CreateObject("Matlab.Application")
Result = MatLab.Execute("surf(peaks)")
```

The Execute method takes a command string as an argument and returns the results as a string. The command string can be any command that would normally be typed in the command window; the result contains any output that would have been printed to the command window as a result of executing the string, including errors.

The advantage of the MATLAB COM Object is its small size because it only comprises the functionality predefined in the compiled MATLAB M-Module. Further on its availability adds to its advantages because the use of such a library does not presume any MATLAB installation. The MATLAB Automation Server on the other hand is based on the idea of establishing a connection, e.g., from a Visual Basic host application to a MATLAB instance running on the same PC or to build up a connection via the internet or intranet to a MATLAB Server. The advantage of this technology is its disposability of the complete MATLAB functionality to a COM communicating application. The disadvantage is the resource consuming proceeding to establish a connection as well as the time consuming process of a calculation request. But there is one significant disadvantage of the MATLAB COM Object which scales down the value of this technology for complex mathematic problems. The current release 13 of

MATLAB does not support the Symbolic Tool Box functionality in the COM builder applications. This means that the solution of the systems of equations which are based on the MATLAB Solve method cannot be executed and remote controlled via a MATLAB COM Object. For this reason we pick the MATLAB Automation Server technology which represents the more voluminous solution but does not include any restrictions. The user can initiate the kinematic computation by pressing the button “getMATLAB Model”. Hereupon the “compute_Motion_Path” starts to inquire the calculation via the MATLAB Automation Server. This one receives the incoming parameter set and calls up the corresponding functions, see Figure 9-12. At first the “computeInvAngles.m” module sends two enquiries to the “compInvKinChain.m” module to determine the matrix describing the inverse kinematic chain of frame B. The second enquiry comprises the calculation of the end effector position in reference to frame A. Both methods make use of the support function “rotationMatrix.m” to complete their rotation computation. The result is linked to the “computeInvAngles.m” function. Here the result of the end effector will be converted into coordinates referencing to frame B and the answer will be forwarded to the “compute_Motion_Path.m” method. This circle will be passed trough for any multiple of the initial angle until the twist range is reached. The multiple is adapted to the input value of the “step” property.

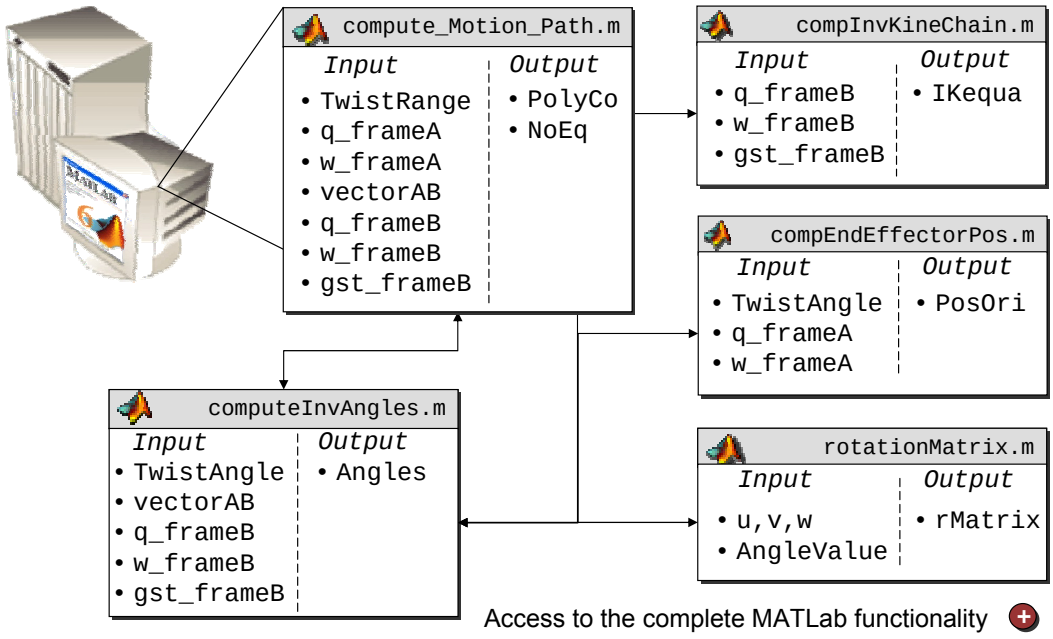


Figure 9-12: Kinematical computation in MATLAB in detail

If all pairs of variates are computed, the curve fitting through these support points is performed. The result of the fitting will be returned to the Visual Basic host application in terms of polynomial coefficients. With the use of these coefficients the actual folding motion can be executed. In combination with a twist angle which gets pretended by the user - the software is in the position to calculate the corresponding values of the constraint angles. Figure 9-13 shows exemplarily the result of such a calculation for $\Theta_1 = 50^\circ, \Theta_1 = 140^\circ, \Theta_1 = 180^\circ$.

The idea of this approximation approach results from the fact that an exact calculation (for each predetermined twist angle) which is combined with several queries to the MATLAB Automation Server would be so time-consuming that a homogeneous visualization of the folding motion is impossible.

But with the use of this curve fitting approach and the specification of the twist range there is only one kinematical computation necessary. The return value, i.e. the polynomial coefficient, describes the folding behavior over the complete fold for any arbitrary angle position inside this range. Thus the MATLAB Automation Server in combination with this approximation method represents an adequate technique to permit a suffieciently accurate visualization, see Figure 9-13.

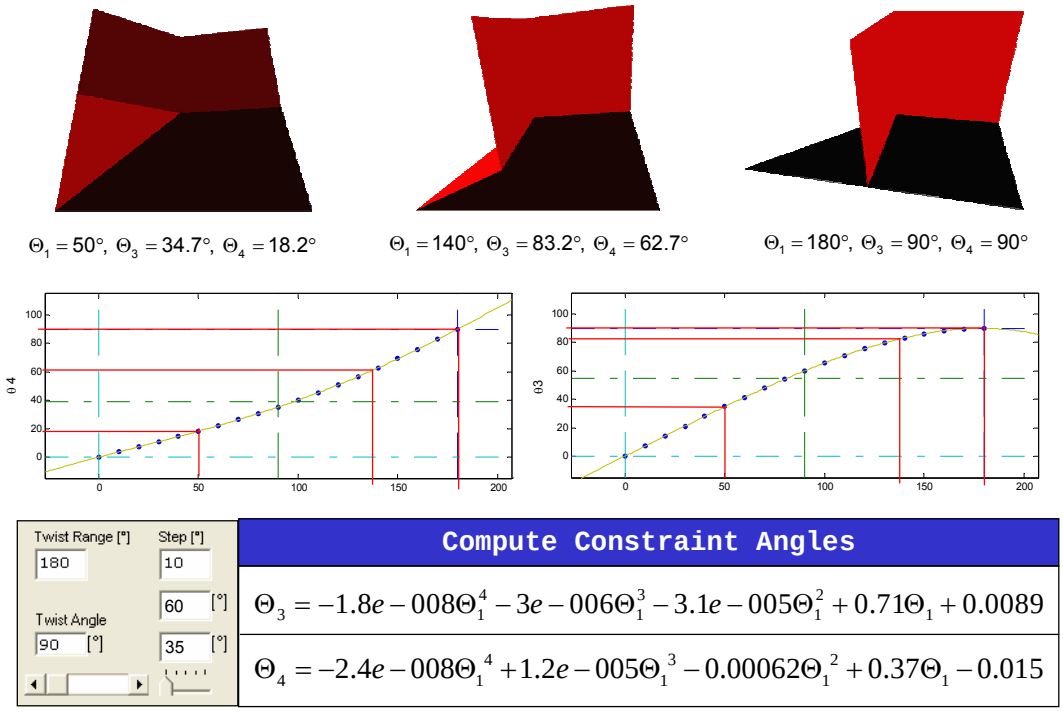


Figure 9-13: Folding motion of the first fold

9.6 Graphical Visualization with the use of OpenGL

The visualization architecture of the folding process is directed to the functional principle of OpenGL. OpenGL is a software interface to graphic hardware. This interface consists of 250 distinct commands to specify the objects and operations to produce interactive 3D applications. OpenGL is designed as a streamlined, hardware-independent interface to be implemented on many different hardware platforms [WND+ 99]. One of the most important characteristics of OpenGL is that it does not provide high-level commands for describing models of 3D objects. This means the development of a graphic model is built up from a small set of geometric primitives – points, lines and polygons that are specified by their vertices.

In order to render a graphic scene consisting of a number of polygons, its object coordinates need to be determined/computed first. The object coordinates comprise the points that describe the shape of the model in 3D space. In order to draw these objects on a flat 2D computer screen, an additional transformation of these coordinates into 2D screen coordinates is required. This proceeding is executed in the so called OpenGL Geometric Pipeline and is illustrated ongoing. First of all we will focus on the creation of the object coordinates of our folding model.

In order to display the geometrical shape of a folding model, we primarily make use of the two primitive geometric objects, a triangle and a quadrilateral polygon. A composition of these objects builds up the folding model. The necessary information describing their shape will be delivered as a dataset of the Kinematic Frame database. The rotation angle of each element relative to its forerunner or back runner is as well known in terms of the twist angle and the computed constraint angles for each point of time during the folding operation.

The folding operation itself can be characterized as a hierarchical drawing problem. If we consider the inverse kinematic part of frame B, e.g., then we determine a simple hierarchical structure. The green element of frame B represents the first element in the structure whereas the yellow one describes the second, see Figure 9-14. A hierarchical structure means in this context, that a rotation of the first element bears as well a rotation and translation of the second. In a mathematic point of view we have to apply the matrix multiplication of $M_3 \cdot M_4$ to element two to describe its current position in space. One has to take into account that the order is important in which the modeling transformations are executed.

This matrix multiplication is implemented in OpenGL by using a matrix stack. A matrix stack is an efficient way to track matrices while traversing a transform hierarchy [Con 03]. This is particularly useful when you are drawing complicated objects that are built up from simpler ones as well as to save and restore certain transformations.

At any point in a drawing, there is a current matrix on top of the matrix stack, composed of all transformations thus far. In this particular example the matrix transformation M4 will be set to the first current matrix stack. The current matrix stack (M4) will be defined with the OpenGL command pushMatrix and canceled with the according popMatrix order. All geometrical transformations between these two operations will be executed in reference to the current matrix stack (M4). This technique allows to determine the object coordinates of each element of the folding model in 3D space by applying the precompiled angle positions to the corresponding matrix stack.

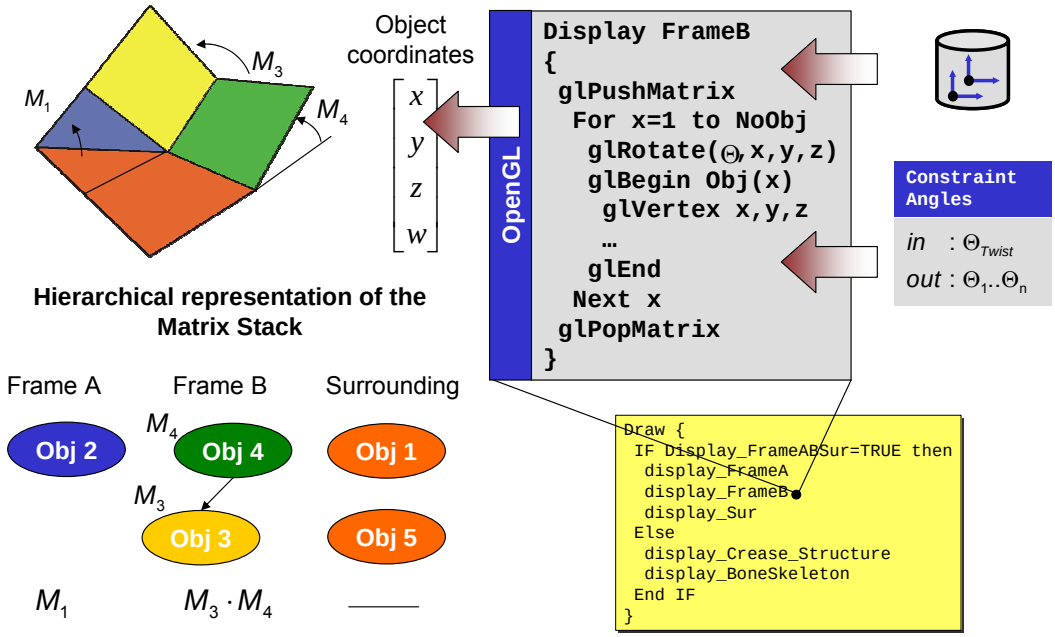


Figure 9-14: Computing the object coordinates

After having been able to compute the object coordinates for each element we explain how to instruct OpenGL to draw the geometric model. For the general understanding of the OpenGL Geometry Pipeline, the analogy of a photo camera is useful.

If you would like to take a picture, you have to position and aim a camera then you have to position and orient the model. All these factors determine what image appears on the screen. OpenGL combines this viewing and modeling transformation into a modelview matrix, which is applied to the incoming object coordinates to yield eye coordinates. These transformations orient the model and the camera relative to each other to obtain the desired final image.

After that you have to choose the lens of the camera. You have to specify what the field of view or viewing volume is and therefore what objects are inside it and to some extent how they look. This is equivalent to choosing among wide-angle, normal, and telephoto lenses, for example. OpenGL applies the projection matrix to yield clip coordinates. This transformation defines a viewing volume; objects outside this volume are clipped so that they are not drawn in the final scene.

The perspective division is performed by dividing coordinates values by w, to produce normalized coordinates. Both the projection transformation and the viewport transformation determine how a scene is mapped onto the computer screen. The projection transformation specifies the mechanics of how the mapping should be performed, and the viewport indicates the shape of the available screen area into which the scene is mapped. Finally, the transformed coordinates are converted to window coordinates by applying the viewport transformation. The viewport transformation explains how to control the conversion of 3D model coordinates to screen coordinates, see Figure 9-15. A manipulation of the dimension of the viewport causes the final image to be enlarged, shrunk, or stretched. Once all the necessary transformations have been specified, you can draw the scene or take the picture.

This procedure will be passed through for each point of time and ensures a consistent graphical representation of the active dataset (Kinematic Frame database) and its corresponding angle position on the screen.

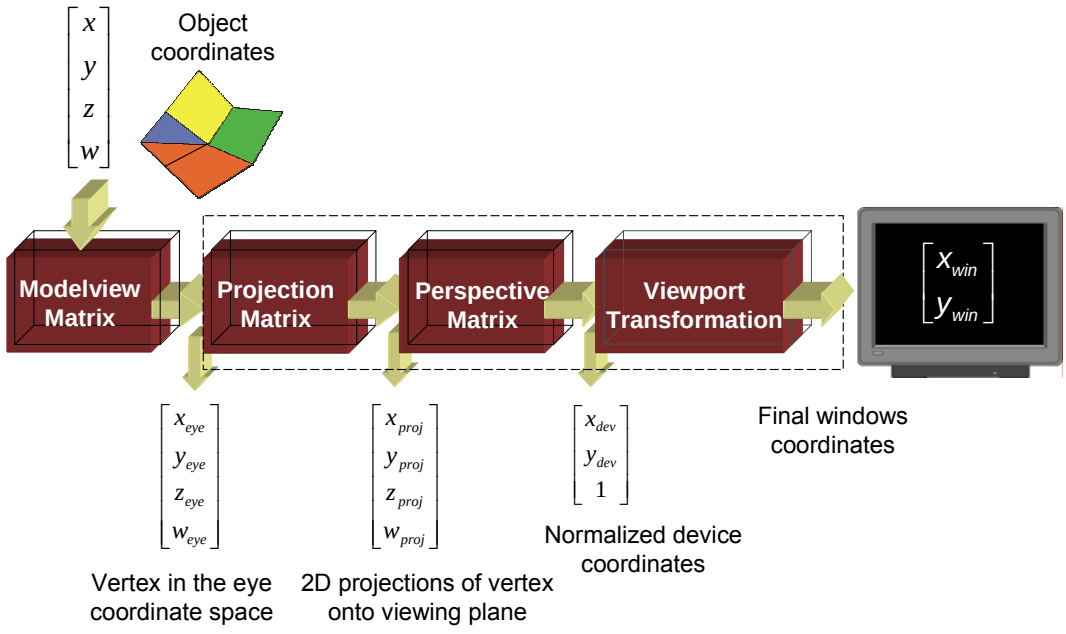


Figure 9-15: Functional principle of the OpenGL Geometry Pipeline

9.7 Realization of the Folding Manager

So far the implementation of the visualization of one fold is completed. As mentioned before, to ensure the transformation from a 2D initial state into the desired 3D shape we are interestied in displaying several folds in a row resp. to repeat selective sections of one fold upon request. This required functional specification is implemented in the Folding Manager.

The Folding Manager can be addressed in the menu “Option”. After the first fold has been completed, the user presses the button “StoreCurrentGeometry” and generates a new starting position to trigger any other fold. The procedure “restoreGeo” calculates the current position of each element participating in the folding in 3D space. Further on it stores the new position in the Geometry database and generates on the basis of this data a new bone structure frame, see Figure 9-16. Hereto the position of each point of the folding model needs be multiplied with a corresponding transformation matrix which describes its current position in space. In order to compute the matrix multiplication we make use of a self developed MATLAB COM Object, the so called Transformation library which provides the function “computeRotatedPoint” to the “restoreGeo” procedure of the Visual Basic host application.

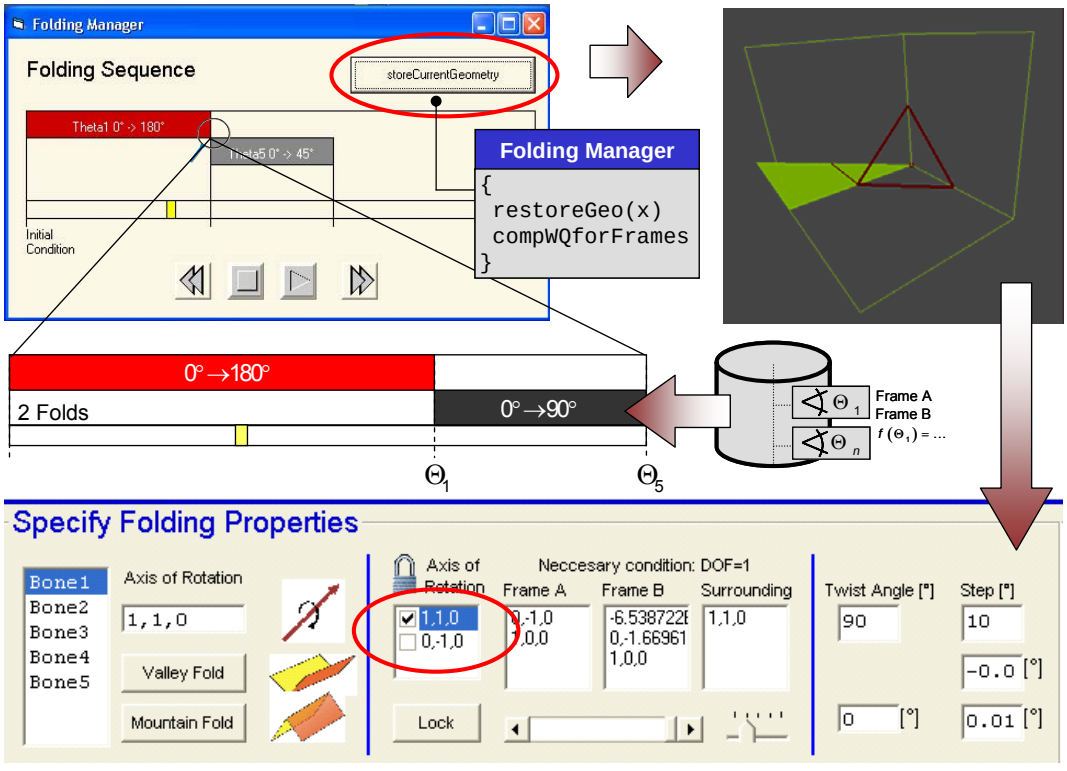


Figure 9-16: Functional principle of the Folding Manager

In this case the MATLAB COM Builder technology represents a suitable solution because of the fact that the required functional range only comprises standard mathematic computations without the use of symbolic expressions. The result of the computation will be used to fill the previously terminated databases with new geometric information about the current position in space, see Figure 9-17.

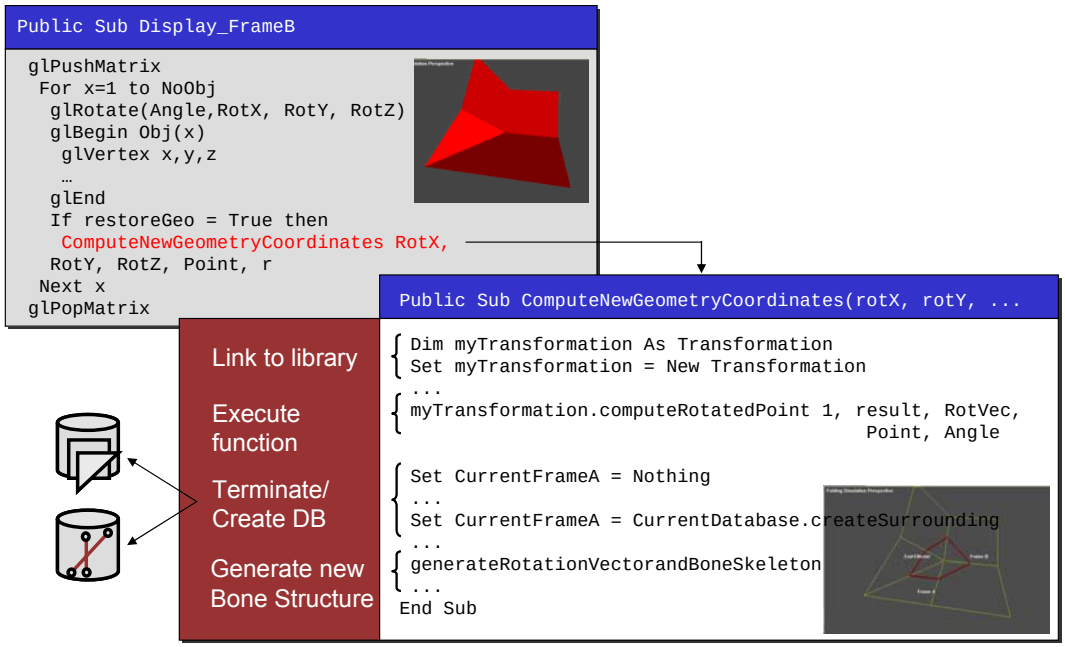


Figure 9-17: Compute current position in 3D space

On the base of the updated data storages, the above mentioned procedure takes place and the user has to specify a new axis of rotation which needs to be locked. Hereupon the “compWQforframes” method generates the corresponding parameters in order to compute the kinematics in the outsourced MATLAB modules. Figure 9-18 illustrates the newly computed parameters after a first fold over $\Theta_1 = 90°$ has been completed. You can extract the corresponding values of q_i and ω_i which build up the base of operation for the second fold.

The realization of the Folding Manager is completed up to this point. The implementation of a slide bar that enables the user to scroll to any arbitrary position inside the complete transformation is still open. There is a need to implement in a further step of development the opportunity to restore any geometric shape on the base of the Recording Motion Database. Thus the main part of the software architecture is realized and its functionality can be comprehended in the following example scenario.

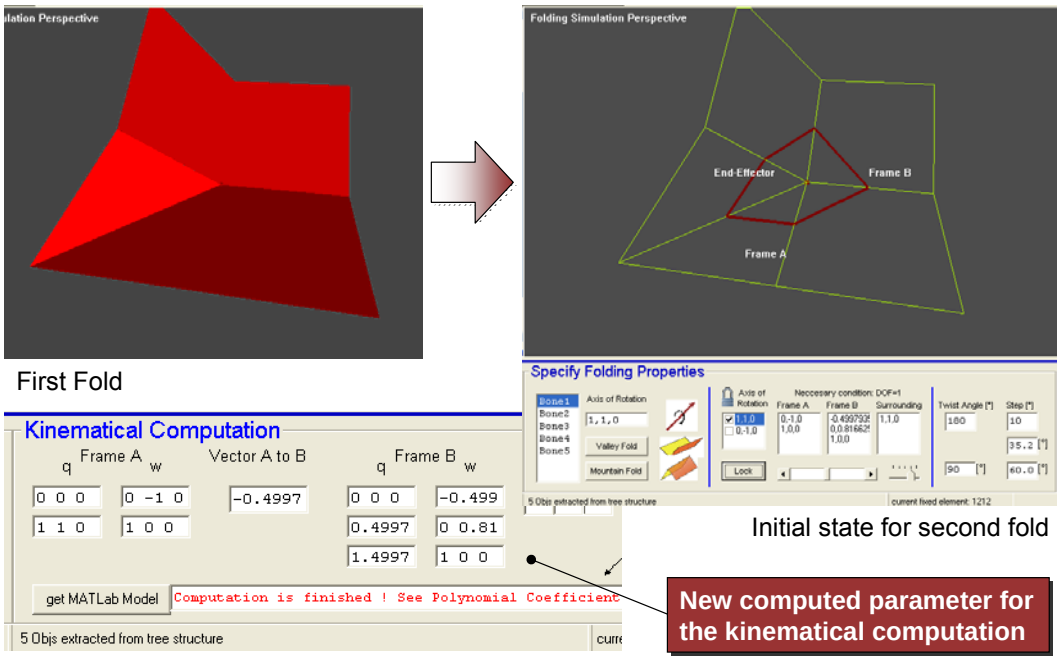


Figure 9-18: Starting position for the second fold

10 Example Scenario

Concluding, the functionality of the software package will be illustrated in an example scenario. The scenario comprises three steps and describes the simulation of an open corner getting folded from its 2D initial state into the intended 3D shape.

At first the creation of the corresponding crease structure will be addressed shortly. Next we will discuss a particular case and try to fold an invalid crease structure. Simultaneously, possibilities will be presented in order to deal with this special configuration. On the base of a valid crease pattern the complete transformation of the demonstration model will be pointed out in detail. This operation is illustrated in terms of the result of the mathematical computation as well as in screen shots visualizing the transformation. At last the multifariousness of the graphic rendering will be demonstrated.

To create a suitable crease structure, which reflects the crease composition of an open corner, the user has to select a combination of precompiled crease patterns in the toolbar menu. The open corner crease pattern consists of one MiddleH pattern that will be applied to the uncreased virtual sheet of paper. Then the lower and upper part are subdivided with the use of the MiddleV pattern and at last the left subelement will be divided with the DiagonalL pattern, see Figure 10-1.

The creation of the crease pattern is only one step in the definition of the crease structure of a folding model. The next stage comprises the definition of the orientation of the rotation axis or in terms of the origami terminology, the alignment of mountain and valley creases. The orientation is one of the determining factors concerning the operability of a folding operation.

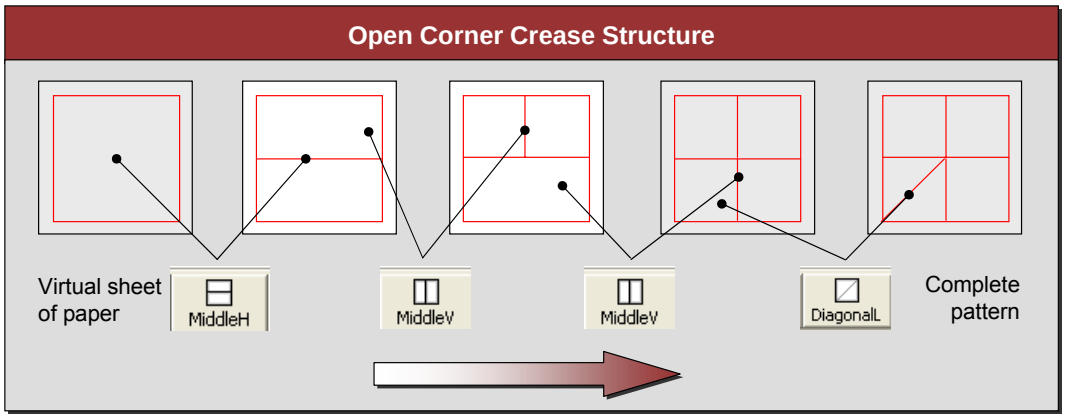


Figure 10-1: Creation of the open corner crease pattern

If a crease pattern and its aligned crease orientation represents an invalid or valid folding structure, ??? can be validated with the result of the inverse kinematic computation. The inverse kinematic chain tries to retrace the end-effector position predetermined by the forward kinematic part. For special compositions this means there is no solution available which matches the end of the inverse kinematic chain with the precompiled end-effector position. Figure 10-2 illustrates the case differentiation into invalid and valid crease structures. The composition differentiates itself only in the orientation of the first axis. The invalid structure consists of a negative oriented axis. Therefore the forward kinematic tries to fold in negative direction whereas the orientation of the inverse kinematic part only allows positive angle positions. Thus this crease composition is invalid. The software informs the user and proposes to review the crease structure.

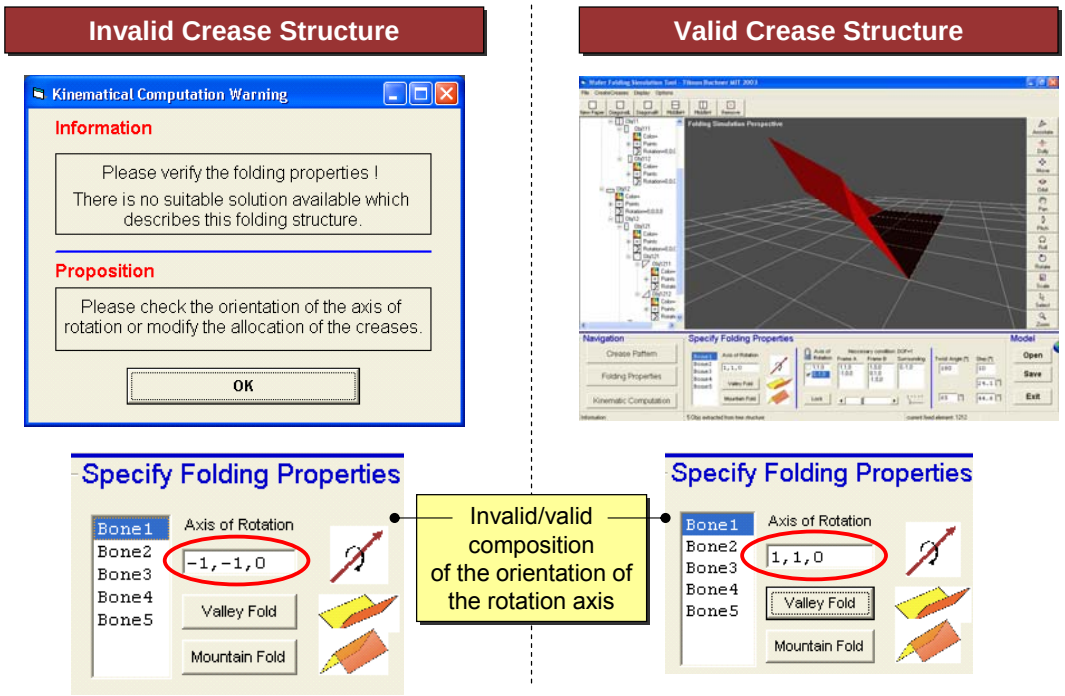


Figure 10-2: Verifying crease structure

If we change the orientation of the first axis into a positive orientation, the crease structure becomes valid. Then the kinematical computation leads to a suitable solution which can be visualized. Figure 10-3 shows the complete transformation from a 2D initial state into the final 3D shape on the basis of a valid structure. Here you can retrace both folds which are necessary to transfer the unfolded open corner into the intended 3D shape. The kinematical computation delivers the transfer functions of the first and the second fold.

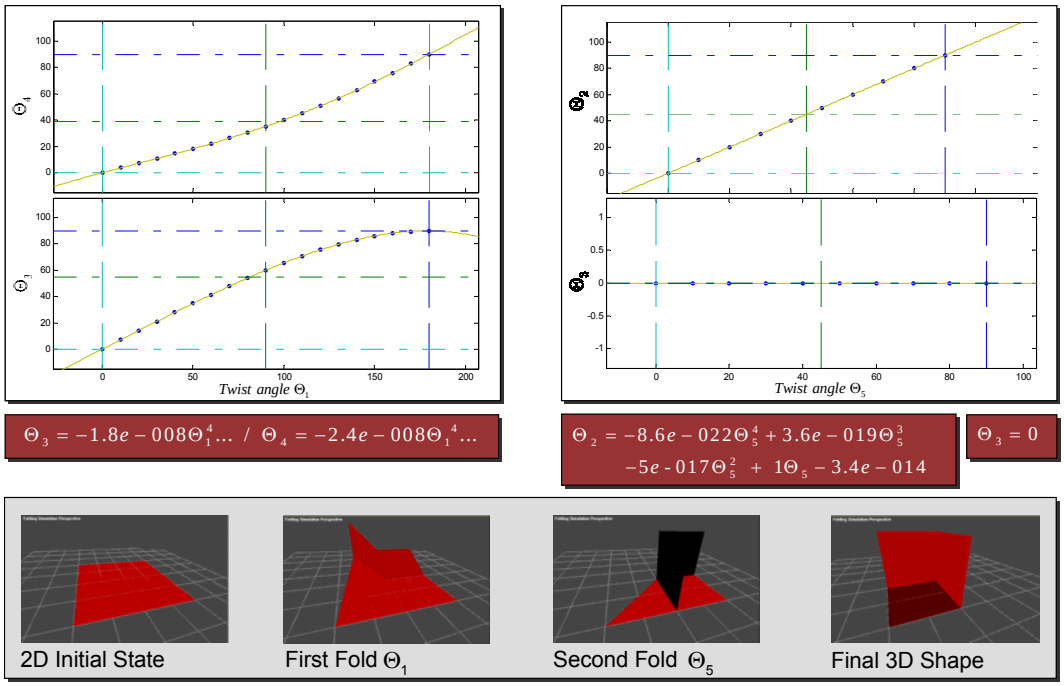


Figure 10-3: Complete transformation from 2D into 3D

On the base of this system of equations the folding motion can be displayed as one can see in the images below the diagrams.

Once the kinematical computation is completed, the user is able to retrace the folding motion in the graphic output. Here the model can be viewed in any arbitrary way. With the use of the graphic tool bar the user has the possibility of rotating the model about the three axes (see c,d) as well as to zoom in and out (see a,b).

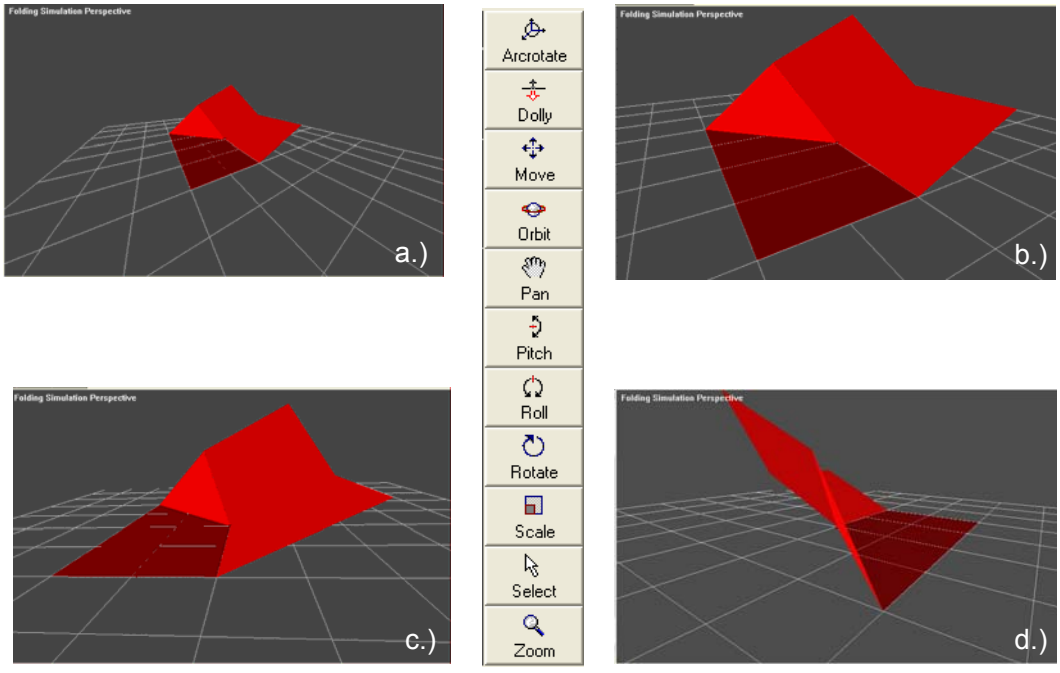


Figure 10-4: Free choice in the view of the folding motion

The opportunity to observe the model from all sides, from the top, from behind or from the bottom of the model, supports the comprehension of the folding motion. Especially concerning typical engineering tasks, this photorealistic visualization contributes to retrace the construction. Finally, the fading of a grid relieves the orientation in 3D space by rotating the model.

11 Conclusion

This thesis introduces a new approach to model origami structures as 3-D polygonal linkages. The approach combines essentials of origami mathematics with already established achievements in the field of robotics and gearing technology and includes a well-known mathematic method (Screw calculus theory). This combination turned out to be a very useful method to describe the kinematics of folding operations.

The result represents a new approach based on the idea to assign a bone structure to the folded segments in terms of a substituted mechanical system. In a further step the degree of freedom of the bone structure will be reduced to one by applying three major constraint conditions (fix one element to the ground, lock axis of rotation, adjust twist range) to perform a unique fold. After having determined these conditions, an overconstraint mechanism comes into existence that can be broken down into two separate systems, one forward - and one inverse kinematic section. At this point, the problem is attributed to a well-known task in the area of robotics which can be solved with conventional inverse kinematic solving techniques. This achievement, reducing the complexity by degrading the DOF as well as subdividing the mechanism into two separate systems, represents the main benefit of this thesis.

In parallel with these theoretical investigations, we have designed and built a comprehensive software tool which allows the user to define an origami structure graphically. The software then translates the structure to a data representation, defines the associated close bone structure and starts simulating the kinematics. As the results of the example scenario show, the theoretical considerations as well as its implementation into a software tool, turned out to work very well. We have achieved folding an open corner demonstration model in two consecutive folds from its 2D initial state into the intended 3D shape.

To put it into a nutshell this newly developed software tool with its kinematic functionality contributes to expand the field of activity for the Nanostructured Origami^{TM} fabrication process. Different folding configurations can be played through with ease. Cost and time consuming efforts to manufacture prototypes of suitable crease compositions can be reduced significantly. In a further step of development, the implementation of self collision detection functionality is desirable as well as the provision for the dynamics of folding objects. The component based architecture contributes

outstandingly to this development. Completing the functionality range, it is of major interest to include the inverse case in future which means the unfolding of polygonal linkages by implementing Ileana Streinu's pseudo triangulation approach.

Finally we will give a short example of a possible area of application for this new manufacturing process for future defense systems. One idea which will be pursued by the 3D Optical Systems Group at MIT is to build up a reconfigurable soldier identification tag (RSIT) that is able to deliver a general-purpose status of a soldier's identification and condition available upon request. The miniature device of sub-mm^3 volume can be mounted on a soldier's breast pocket as a pin, or resp. on his helmet or backpack. To request his status, a low-power, eye-safe laser beam is used to retro-reflect the soldier's identification modulated on the reflected wavefront, see Figure 11-1.

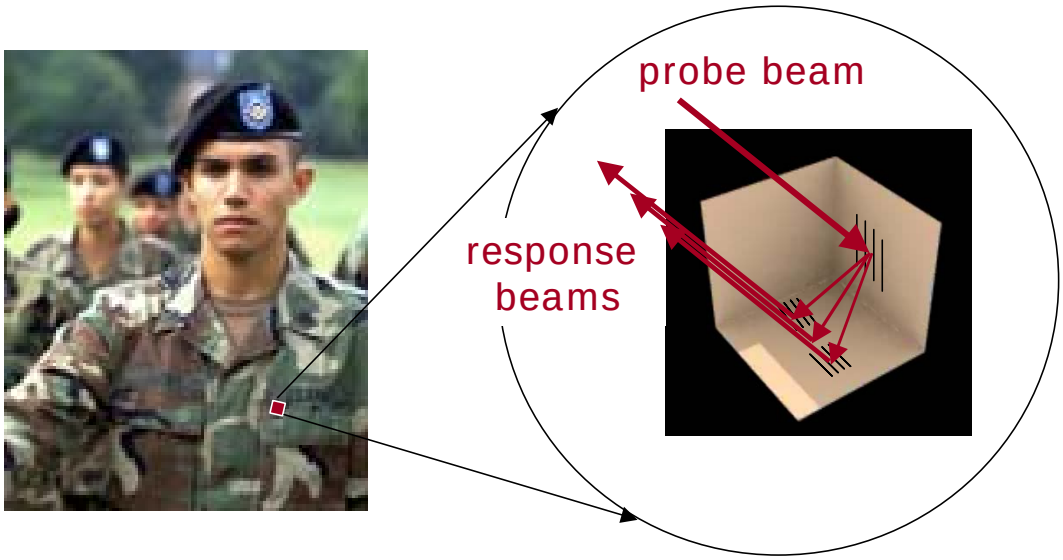


Figure 11-1: Reconfigurable soldier identification tag

The operation of RSIT is based on the open corner demonstration model with one additional specification - a diffractive optical element (DOE) patterned on the inner surfaces. The DOE imprints the identification information on the retro-reflected wavefront. The geometric shape of the open corner contributes to the fact that the RSIT can be read out from virtually a 2π solid angle. The design of the device can be arranged to retro-reflect the correct information only when probed by a specific wavefront. This feature is called key-based operation and is used to respond only to a specific wavefront modulated on the probe beam, whereas if an adversary attempts to read the tag without using the correct key, the adversary will simply receive a diffusely reflected beam without any meaningful information.

For this purpose, the DOEs are patterned via lithography on the flat surfaces during the 1st step of the Nanostructured Origami™ process. In a second step of manufacturing, the RSIT folds itself into the intended position which results in high yield and low cost. This feature enables as well the reconfigurability during the operation. This means that the RSIT can fold itself to change its operation, e.g., to flip itself around to retro-reflect confusing information to adversaries.

This example illustrates outstandingly the technological benefit of the Nanostructured Origami™ technology. The determining factor of this application is the folding mechanism. Other by *folding*, gratings cannot be fabricated on “vertical” walls with conventional lithography! To obtain a retro-reflector with DOE, as shown in Figure 11-1, the *only* other alternative would be to manually assemble discrete elements, which would be expensive and prone to inaccuracies and low yield. Thus, the Nanostructured Origami™ technology in combination with the presented achievements concerning the folding of polygonal linkage opens up new vistas for a wide range of future applications in both military as well as commercial arenas [Bar 03].

12 Abbreviations

2D	Two Dimensional
3D	Three Dimensional
API	Application Interface
CC	Crease Collection
CG	Crease Generator
COM	Component Object Model
DARPA	Defense Advanced Research Projects Agency
DB	Database
DNA	Deoxyribonucleic Acid
DOE	Diffractive Optical Element
DOF	Degree of Freedom
FK	Forward Kinematic
GUI	Graphic User Interface
ID	Identification Code
IK	Inverse Kinematic
MATLAB	Matrix Laboratory
MBS	Multi Body System
MCLS	Multi Close Loop System
MEF	Matrix Exponential Formulation
MEMS	Micro Electro Mechanical System
MIT	Massachusetts Institute of Technology
RSIT	Reconfigurable Soldier Identification Tag
TTS	Tree Type System
VLSI	Very Large Scale Integrator

13 References

[AmSo 00] *Amato, N.; Song, G.*: Motion Planning Approach to Folding: From Paper Craft to Protein Structured Prediction, Technical Report 00-001 Department of Computer Science Texas A&M University Jan 17,2000

[AoId+ 03] *Aoki, K.; Ideki, T., et. al.*: Microassembly of Semiconductor Three Dimensional Photonic Crystals, Nature Materials, Vol. 2, February 2003

[Barb 03] Conversation with: Prof. George Barbastathis, 3D Optical Systems Group, MIT 2003, date: 05. Oct. 2003

[BeHu 02a] *Belcastro, S.; Hull, T.*: A Mathematical Model for non-Flat Origami; Origami^3, The Third International Meeting of Origami Science, Mathematics and Education published by AK Peters, 2002

[BeHu 02b] *Belcastro, S.; Hull, T.*: Modelling the folding of paper into three dimensional using affine transformations, Linear Algebra and its Application 348 (2002) pp.273-282

[BJH+ 03] *Barbastathis, G.; Jurga, S.; Hidrovo, C.; Niemczura, J. Et al.:* Nanostrucutred Origami, IEEE-Nano 2003, August 12, 2003

[BTC+ 97] *Bowden, N.; Terfort, A.; Carbeck J.; Whitesides, G.*: Self-Assembly of Mesoscale Objects into Ordered Two-Dimensional Arrays, Science, Vol. 276, April 1997

[Con 03] Controlling the Order of Transformations: *GL3.2 Version 4.1 for AIX: Programming Concepts,*
http://as400bks.rochester.ibm.com/doc_link/en_US/a_doc_lib/aixprggd/gl32prgd/cntrltransfrms.htm
publication date: -, released order date: 15 Dec. 2003

[Dema 03] *Demaine, E.D.*: Folding and Unfolding Linkages, Paper, and Polyhedra, Department of Computer Science, University of Waterloo

[Diet 03] Conversation with: Univ.-Prof. Dipl.-Ing. Dr.techn. Peter Dietmaier, Institut für allgemeine Mechanik, Technische Universität Graz,
date: 28. Oct. 2003

[Hull 94] *Hull, T.*: On the Mathematics of Flat Origamis, Congressus Numerantium, Vol. 100 (1994) pp. 215-224

[Hull xx] *Hull, T.*: Counting Mountain-Valley Assignments for Flat Folds", Ars Combinatoria, Department of Mathematics, Merrimack College, North Andover, MA 01845

[Huzi 03] *Huzita's axioms*, Wikipedia the free encyclopedia
http://en2.wikipedia.org/wiki/Huzita's_axioms
publication date:17 Jun. 2003, released order date: 19 Nov. 2003

[Huzi 92] *Huzita, H.*: Understanding Geometry through Origami Axioms, Proceedings of the First International Conference on Origami in Education and Therapy (COET91), J. Smith ed., British Origami Society, 1992, pp. 37-70

[IsOH 98] *Isao, S.; Osamu, K.; Hirofumi, M.*: 3D Micro-structures Folded by Lorentz Force, The Eleventh Annual International Workshop on Micro Electro Mechanical Systems, 1998

[Jurg 03] *Jurga, S.*; 3D Micro and Nanomanufacturing via Folding of 2D Membranes, Master thesis MIT, June 2003

[Kang 00] *Kang, T.G.*: Solving Inverse Kinematics Constraint Problems for Highly Articulated Models, Technical Report CS-2000-19, Waterloo, Ontario, Canada, 2000

[Lee 02] *Lee, T.H.*: A Vertical Leap for Microchips, Scientific American, January 2002

[Math 03] *The MathWorks, Inc*: What is MATLAB ?
http://www.mathworks.com/access/helpdesk/help/techdoc/learn_matlab/learn_matlab.shtml
publication date: 2003, released order date: 10 Dec. 2003

[Matr 03] *Matrix Semiconductor*, http://www.matrixsemi.com/3dtech.shtml?26, released order date: 05 Dec. 2003

[Micro 98] *Microelectromechanical Systems (MEMS)*, Commerce Business Daily, Publication Date: November 6, 1998; Issue No.: PSA-2217

[Mitc 03] *Mitchell, D.*: Origami Heaven, http://www.origamiheaven.com
released order date: 01 Dec. 2003

[MrLS 94] *Murray; Li; Sastry*: Mathematical Introduction to Robotic Manipulation, CRC Press1994

[Nagp 00] *Nagpal, R.*: Origami Programmable Cell Sheet, Amorphous Computing Group, MIT Artificial Intelligence Lab, Dec. 2000

[Open 03] *OpenGL.org: OpenGL - The Industry's Foundation for High Perforance Graphics, http://www.opengl.org/about/overview.html#1*
publication date: 2003, released order date: 10 Dec. 2003

[Pal 96] *Palmer, B.*; Tree Rings
http://www.conservation.state.mo.us/conmag/1996/oct/7.html
publication date: Oct. 1996, released order date: 24 Oct. 2003

[Reed 03] *Reed, D.*: Consumer electronics drives chip design technology, EEdesign, 18 Jul. 2003

[Schm 02] *Schmolesky, M.*: The Primary Visual Cortex
http://webvision.med.utah.edu/VisualCortex.html
publication date: Dec. 2002, released order date: 01 Dec. 2003

[Sent 01] *Senturia, S.D.*: Microsystem Design, MIT, KAP 2001

[Stre 03] *Streinu, Ileana:* A Combinatorial Approach to Planar Non-colliding Robot Arm Motion Planning, Department of Computer Science, Smith College, Northampton, MA USA

[Tang 00] *Tang, C.W.*: DARPA MEMS PROGRAM, Vision statement
http://www.darpa.mil/mto/mems/vision/index.html,
publication date: Jan. 2000, released order date: 02 Nov. 2003

[Tris 03] *Tristram, C.*: The Transistor Reinventing, MIT Technology Review 09/2003 p. 56

[Weiss 99] *Weissstein, E. W.: Origami, Wolfram Research, 1999 CRC Press LLC* http://mathworld.wolfram.com/Origami.html publication date:1999, released order date: 19 Nov. 2003

[WND+ 99] *Woo, M.; Neider, J.; Davis, T.; Shreiner, D.:* OpenGL Programming Guide, Third Edition, Addison-Wesley, 1999

[ZhJS 99] *Zhang, X.; Jiang, X.N.; Sun, C.*: Micro-Stereolithography of Polymeric and Ceramic Microstructures, Sensors and Actuators 77, 149-156, 1999.