Bionics in Architecture

Experiments with Multi-Agent Systems in Irregular Folded Structure

Diploma Thesis by Benjamin Felbrich
submitted at

Deptartment of Architecture - Juniorprofessorship of Knowledge Architecture
Jun. Prof. Dr.-Ing. Joerg-Rainer Noennig
Tutor: Sebastian Wiesenhütter

Department of Mathematics - Institute of Geometry
Prof. Dr.-Ing. Daniel Lordick

Technische Universität Dresden

Content

A Summary

This work attempts to find a way to apply the operation mode of highly collaborative multi-agent organisms, such as ant colonies and termites to tasks of architecture and building construction. The strategies of these animals enable them to fulfill sophisticated tasks by interacting in both physical and notional manners; for example, collectively lifting heavy objects (physical) or finding the shortest paths to food sources (notional). Their colonies consist of very simple, similar individuals that do not act through a central control, but inter-dependent. To tap the full potential of these strategies, it is necessary to translate them in a way so that we can apply tools of our scope such as computer simulation, prototyping and coding. The mode of this translation is logic and geometry.

At first the hereby submitted thesis identifies a suitable and sufficiently complex issue in an architectural context: the generation of free form shapes with a large amount of interconnected agents. From a desired real world applicability derived the preset conditions of standardization and simplification in the units. In order to realize this task, different tools were utilized: Initially a lot of simulations were done to investigate on the physical behavior of such a system, the way signals would spread and how a possible communication system would look like. In order to achieve results that provide undoubted validity in the real world and, in contrary to the simulations, don't remain speculation it was necessary to introduce another method: research through design - prototyping. By actually building this kind of robotic agent one could investigate on its mechanical properties and the informational system it would need to work with. To do so an exemplary free form surface was created and appointed to be regenerated with simple interconnected actuators.

The design of the respective agent was refined in three consecutive prototypes, where the knowledge gained from the antecessor was utilized to improve the descendant. As a result, the third prototype showed the mechanical feasibility of its design. It supplies sufficient power to lift itself and its neighbors off the ground. Furthermore it provides sensors that are capable of giving a feedback of its actual state. Its control coding, however, remains unfinished in the sense of an intelligent, merely independent behavior. At the point where these lines are written, it is still not able to operate without a central control or a profound, previously implemented knowledge of what to do, represented by preset control curves. The code, however can still be refined. In this sense, the physical model can be seen as a container, an embodiment of a logical system or a field for experimentation where we can test different programmings.

1. Introduction

In nature, solutions for complex issues evolved over millions of years. From sulfur-reducing bacteria to the famed ever stretching giraffe neck, nature repeatedly succeeds to overwhelm us with its adaptability to various environmental conditions. In biology we established the science of observing and trying to understand the mechanisms behind to disentangle the clutter of its microscopic interlocked processes. The human being of course, since it is a biologically functioning entity, is of tremendous interest to these investigations as well.

We as humans rationalize and empirically describe the world we live in through mathematics and geometry. In these sciences we developed a way to abstract, understand and communicate logic mechanisms and causal chains our environment is based on. Furthermore, we establish ideas, create products and buildings using sophisticated technology derived from that knowledge. Thus, the combination of these three fields is not a contradiction at all, but rather a logical conclusion. Using the tools of mathematics and technology to research about principles found in nature represents a valid method of operating, just like, vice versa, the use of biological principles to enhance our instruments - bionics. If we conceive architecture as a product of technology, to be optimized, to be describable in an empirical manner, to *function* we can take big strides by utilizing the trinity described above. This definition of architecture, however, fails miserably in taking into account and expressing the very core need of man to descry beauty and delight, to enchant and to be enchanted by art. The thesis submitted herewith isn't able to cope with the artistic aspects of architecture; generally the capabilities of a scientific report's scope to do so remain questionable. But just as nature produces stunning beauty by simply solving problems at hand without a bigger plan in mind, with the absence of an immanent *intention* to do so, maybe even a construct, physical or notional, simply designed to deal with a dry logical problem can evolve into something we perceive as beautiful. The meandering, incomprehensible shapes of the Mandelbulb come to mind, or, most classic, the Finobacci sequence.
The core issue, however, this work is rather dealing with is of pure rational kind: Which improvements in the construction process can we gain from utilizing bionic principles apart from mimicking shapes and mechanical, *physical* operation modes? Which adaptations in abstract logical causal chains can we identify and utilize to optimize structure? Investigations on complex dynamic systems as ant colonies, termites or bees have shown, that it is not necessary for a collaborative organism to consist of individuals of high complexity in order to achieve smart behavior such as the erection of sophisticated, highly integrated built environments. Merely their interaction and sheer mass eventuate in workings of high functionality. Some of these principles are already in use in other fields, like the well known ant colony optimization algorithms. Discovered by Marco Dorigo in 1992 [1], the logical mechanism of ants utilizing their environment as an envelope to communicate indirectly was translated into an algorithm. Nowadays this strategy is used to tackle intricate problems like route calculations in navigation systems. Biologist and Stanford Professor Deborah Gordon greatly intensified the research in this field, unveiling the complex organization of the ants' state, the categories of its individuals the different communication modes and their benefits for the species[2]. The allocation of tasks in absence of a central control or even a global informative network, but merely as the result of the most rudimental individual communication plays a central role in her research.
In an architectural context the issue had been dealt with to a large extend on a different scale. There are various examples of *process* oriented urban planning which bypass the classic approach of providing a ready-made urban design *product* to be tamped onto cities like Indore, Caracas or Amsterdam. It merely seeks to

answer the question, how small interventions in the daily live of its inhabitants can trigger desirable mechanisms in the dynamic system of the city, finally leading to a better living. Does a gymnasium in the midst of Caracas' favelas transform the city better, than a new master plan, attempting to extinguish this area would do? [3] Does the plain offer of absolutely simple plumbing units, instead of expensive entire living units, along with the permission to add building parts as desired create an atmosphere of responsibility and thereby an even cleaner city? It does [4]. Can we even *gamify* and thereby directly outsource the process of urban design to the cities own inhabitants? [5] Though the parallels to ant states are not obvious, they do exist: The power of many can, if canalized properly lead to unexpected improvements of living standards and reacts much more flexible to changing conditions. And it doesn't need much to do so. The tools to regulate and trigger these developments are rules and standards. In ant states it's the rules on when to leave a pheromone mark and how to interpret it. In urban planning it is the *policy* regulating the interaction of the individuals.
If we abstract the principle of this operation and apply it to the lower scale of single buildings, we can rise the question, how collaborative interaction of many can help to construct a building or to optimize its construction in certain aspects. And we can further apply some of the important properties of the previous examples to the new scale: none of the agents has to have knowledge of a higher goal, they act interdependent, but without a direct central control, they do not necessarily intend to change their environment, but follow simple rules. Yet, aside from that in the lower scale we have to identify new *tools*. This work is trying to find one of them.
Lastly the thesis hereby submitted is an experiment with a different mode of operation. In [6] Gesche Joost describes her idea of research through design and the necessity of physical prototypes to gain knowledge in a design context. After weeks of working in the mode of pure thought and simulation with moderate success, the idea to test and try this approach in a physical model seemed very adjuvant. In addition, due to the way architects usually work, by elaborating physical models to depict ideas and to prove or falsify their feasibility, represents the most natural working mode. Only the problem to investigate on is a rather unusual one. In that sense, this is just a test.

2. Concept

2.1 Basic Idea

To design buildings using free form shapes has been a common manner among architects all along. However, its actual realization in building construction has been quite a complex objective to tackle and oftentimes represented an insuperable hurdle on the way from an architect's sketchbook to built environment. Nature though shows us, that it is possible to utilize the power of many, relatively simple agents of a species to reach goals far beyond the capabilities of a single entity. The key is collaboration. What if we could utilize this aspect in erecting structure? In order to apply the principles of interaction, as they are discussed in the introduction to a small, architectural scale, it was attempted to break them down to a very specific experiment: *Can we design simple agents, to make them collaboratively regenerate a given free form surface?* How would these agents have to look like? What are the minimum capabilities they need to provide to reach their goal? Do they communicate? And if they do so: How? What are the naturally given constraints to the approach?

Fig. 1.1: Termites' nest [a]

Fig. 1.2: Ants collaborating [b]

There are several constraints though that derive from both, the biological origin of the matter and the applicability in architecture: simplicity in the mechanism, equivalence of the units, interconnection.

The way this issue was approached towards in general is that of an architect's toolkit: establish an idea, model it, prove or falsify, retry. Modeling however can be of both kinds, physical and virtual. Apart from that, I will try to back up the investigations with a geometrical foundation.

2.2 The Agent

The very first step was to find the basic mechanism on which the agents would operate. Several ideas were drawn, discussed and refused. Three of these options were taken into consideration. They are illustrated here to exemplify the fundamental differences and the reasons which led to the final choice.

Option 1: Triangle telescope actuators

A system of interconnected telescope actuators forming a triangular grid represents a very simple system. It combines a large number of agents to a interconnected mesh (the video clip *"01_131023_telescope_mesh.mp4"* in the appendix attached to this work shows its functionality). Their length is variable and can adapt to specific needs the surface to be obtained demands. Its advantages are at hand: it is easy to assemble and provides a high flexibility. Since computer graphics is based on triangular polygon meshes for a long time, there's a large toolbox available from the computational point of view (spring-particle models). A big disadvantage of this scheme in a practical use lies in the complicated nodes. They would have to be able to link a various number of edges and react flexibly towards any movement. One could think of a versatile version of the Mero Ball Node System, a rather complicated detail. Fig. 1.5 shows one quite inaccurate and insufficient option of solving this issue with a ring-like joint.

Fig. 1.3: Option 1: agent

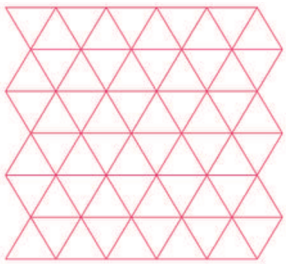

Fig. 1.4: Option 1: topology

Fig. 1.5: Option 1: assembled system

Option 2: Telescope actuators on a non-triangular grid

Derived from early experiments utilizing the logic of papaya stem growth and recovery after damage (appendix: video clip *"02_131010_papaya_trunk.mp4"*), the second system illustrated consists of telescope actuators controlling a hexagonal substructure. Advantageously the substructure bears loads giving the actuators the freedom to only apply a controlling force. Under examination this approach also bears some disadvantages: It is a quite inflexible system. The faces, not being triangular, are not necessarily planar, which complicates the control of this structure a lot. One would always make an effort to keep the facets planar . Furthermore, the direction of its hinges axes aligns with the normal vector of the shape to regenerate, making them inevitably bend and eventually crack in strongly curved areas.

Fig. 1.6: Option 2: agent

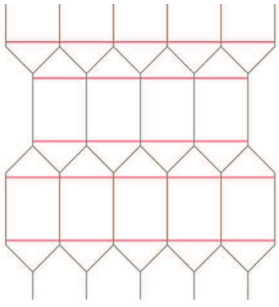

Fig. 1.7: Option 2: topology

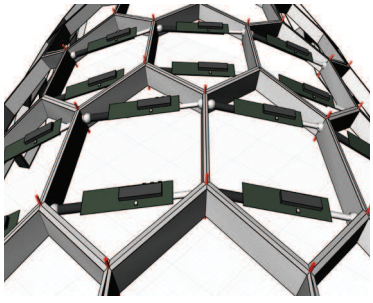

Fig. 1.8: Option 2: assembled system

Option 3: Angle actuators on a triangular mesh

This system of wing-like angle actuators became the most relevant one (appendix video clip: "*03_131023_angle_actuator_mesh.mp4*"). Besides a high flexibility and simplicity in assembling, it provides a variety of shapes to regenerate. Due to its triangular alignment it inherits the advantages in control described in option number one. Furthermore it avoids the biggest disadvantage of the latter: the complicated nodes. If we define the angle between two triangle facets as the only variable to be changed, as this actuator intends to do, the edge lengths of the underlying triangular mesh need to stay invariant. Following from that is, that this options is based on the same topology like option number one but shifts flexibility from the edge lengths to the angle between two facets. This results in a more rigid behavior than option number one. In this respect each have their respective merits. In direct comparison, however, option number three appeared as the more suitable one, due to its much simpler mechanism.

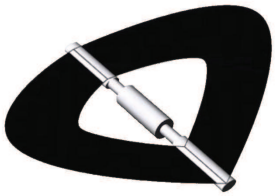

Fig. 1.9: Option 3: agent

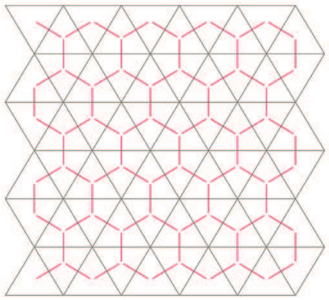

Fig. 1.10: Option 3: topology

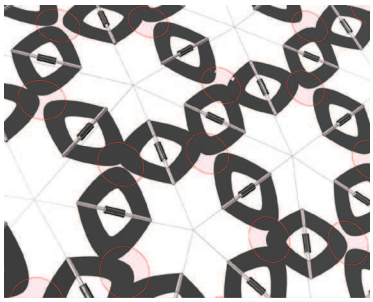

Fig. 1.11: Option 3: assembled system

After deriving this topology and basic mechanical mode from pure practical reasoning a stunning parallelism to another process found in nature was found: folding.

2.2 Folding

In developing leaves and blossoms in a compact, folded state to their almost completion entirely within a bud, plants found a very efficient strategy to save their most fragile parts from the hazards of frosty weather or natural enemies. It is not until these parts have reached a temporary maximum of potential effectiveness in attracting pollinators or collecting energy through sunlight, that, triggered by a certain environmental signal (e.g. increasing temperature during spring time), the plant activates this potential. The blossom or leave unfolds within a short time, revealing its full potential to benefit the plant.

The correlation with architectural scope of duties is obvious: The need to "grow" - or better: to assemble - a structure to a far extend under convenient conditions and "activating" their structural potential - for example a wide span length - on site in the very last steps of edification. Thereby we can tremendously simplify assembly.

This procedure, however, represents a common approach towards construction in general. Even the prefabrication of concrete pieces, wooden framework or any other construction-relevant, pre-assembly operation could be interpreted as a bionic process in this sense.

In actual large scale planning, static, immobile folded structures come into play due to their favorable structural property of combining planar sheets in a way that vertical loads are deflected into the sheets tangent direction, which tremendously increases stability (example fig. 1.13). Due to the nature of our system, these aspect prove irrelevant. On a low scale, however, for example in furniture design or temporary light weighted structures, but also in solar sails of satellites, the actual process of (un-)folding, of parts being moved, becomes feasible, revealing the procedure's capability to illustrate the coherence to nature described above in the most apparent manner. An entirely assembled system of interconnected pieces (mostly planar), changing their dimension and thereby their effectiveness and usefulness through the simple process of unfolding. It is to no surprise that folding structures gain more and more attention among structural engineers and designers. Approaches like the Concrete origami project at RWTH Aachen and TU Dresden shown in figure 1.14 attempt to explore the potential of *un*folding a structure even beyond the scale of furniture design, making use of new, super light concrete compositions. These state of the art examples merely adopt relatively simple, mostly symmetric one-step fold patterns like the Miura or pineapple folds. These patterns are capable of regenerating a limited amount of shapes, mostly flat or curved in one direction. Reconsidering our initial idea to be able to regenerate *any* free form shape the opposite approach seems more feasible: instead of starting with a fold pattern and see which shapes it is able to generate, we'll start from the surface we want to approximate and recursively find its respective fold pattern.
In this respect it makes sense to have a closer look on the geometric foundations of free form shapes and their discretization.

Fig. 1.12: Coreopsis bud unfolding [c]

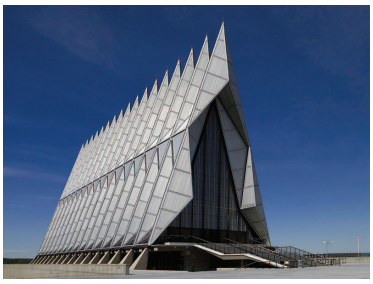

Fig. 1.13: Airforce Academy Chapel. SOM [d]

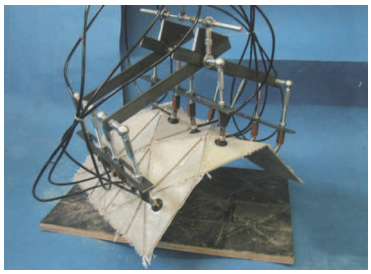

Fig. 1.14: Concrete Origami, RWTH Aachen [e]

3. Geometric Approach And Simulation

The topological abstraction of the described folding setup is represented by a triangular polygon mesh. Since polygon meshes have been a subject of matter in computer graphics for as much as four decades with their geometric and theoretical foundation being much older, a large toolbox is provided if we can translate the issue of folding into an issue of computational geometry in polygon meshes. It is, however, crucial to restrain the variety of possibilities this field of science provides in order to find an alignment suitable to our purpose. As a precondition we'll determine, that we have a given free form surface, elaborated in the design process, which we want to regenerate using a system of simple, physical components. This free form shape is arbitrary which means that our construction kit has to provide the flexibility to adapt to a large variety of shapes put in front.
The main concerns in this sense is a high degree of smoothness in curvature our system should be able to generate. In geometric terms we can rephrase this aspect and say, our system of individual, composed pieces has to be able to generate a large and complete spectrum of curvature (namely Gaussian curvature) in its vertices by folding and being reassembled. The other important concern, due to the intended architectural context lies in a preferably high degree of standardization in the utilized components in oder to minimize complexity in their production process, and thereby minimize cost. Apart from that folding is, in contrast to the pure static architectural context, a dynamic process. Thus we also have to take the flexibility of our desired structure, its maneuverability into consideration.

3.1 Discrete polyhedral surfaces - standardization vs. smoothness

In computer graphics the discrete representation of an arbitrary free form surface can easily be regenerated by subdividing it ad libitum. The smaller the faces get, the higher the resolution and thus, the higher the smoothness. In building construction and especially in our example, however, the will to simplify forces us to cut down the variety of pieces we use. Specifically spoken, we need to limit the amount of utilized edge lengths to provide a simple to produce construction system. This is why we'll start with the most simple layout of a possible triangle mesh, a grid of equilateral triangles (fig.3.1). Six edges meet in one node spanning an angle of 60° each (Fig. 3.1). In [7] Bobenko et al. define the discrete Gaussian curvature K of a polyhedral surface S at the point p as:

$$K(p) := 2\pi - \sum_i \alpha_i$$

Where α_i are the angles between consecutive edges. The points with $K(p) > 0$, $K(p) = 0$ and $K(p) < 0$ are called *spherical*, *euclidean*, and *hyperbolic* respectively. In our first example we can easily calculate the Gaussian curvature over any node:

$$K(p) = 2\pi - 6 \cdot \pi/3 = 2\pi - 2\pi = 0$$

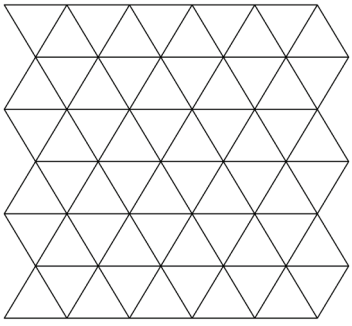

Fig. 3.1: Regular equilateral triangle mesh

Since the components of our system are supposed

to be rearranged arbitrarily, we can add or remove triangles from each node in order to change its curvature:

$$K(p) = 2\pi - t_p \cdot \pi/3$$

Here t_p represents the amount of adjacent triangles. It can't be smaller than three, otherwise we couldn't connect the triangle's ends to close the knot.
Doing this we get a discrete sequence of values for K, depending on t_p. It is important to mention, that this fact only applies, if every neighbor vertex of p also lies on the surface S. In another manner we could also fold the pieces around a node in a way that we don't "count" some of the triangles attached and virtually

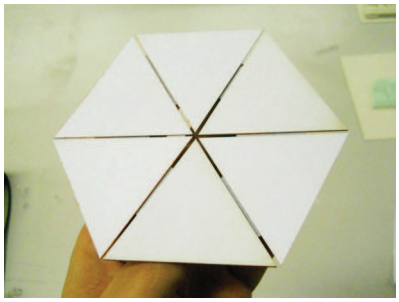

Fig. 3.2: Single node with zero Gaussian curvature

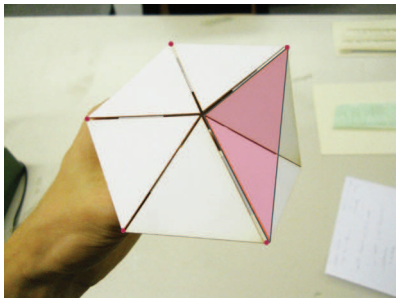

Fig. 3.3: Modifying the Gaussian curvature

bridge the gap they would leave by imagining a single triangle instead of these two (Fig. 3.3 in pink). This means, that we could adjust the Gaussian curvature at this node continuously, without taking steps. In return we get nodes, which do not lie on the free form surface S to regenerate. This ultimately results in an edgy, non smooth appearance and would also negatively effect nodes in the neighborhood of p. So we are facing the dilemma of either achieving a very limited amount of obtainable curvatures or accepting a bumpy appearance with a virtually correct curvature.

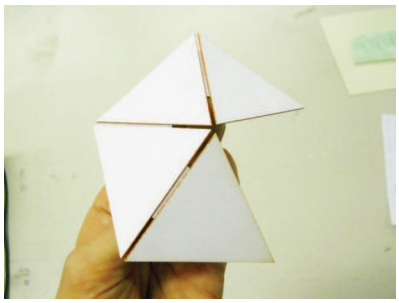

Fig. 3.4: Pieces left out

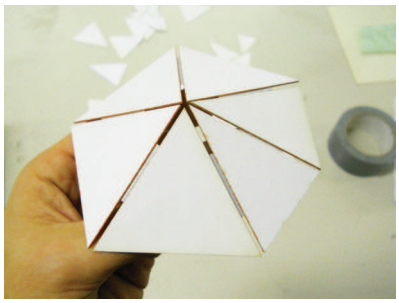

Fig. 3.5: Introducing a different length

There are two possible solutions to this issue:

1. Instead of "not counting" the triangles we didn't want to take into consideration to achieve the desired curvature, we simply omit them (fig. 3.4). If we imagine a much larger structure with more than just one node, this means, that we could perforate the mesh. This also provides the advantage of an increase in flexibility in the process of folding.

2. We allow to make use of another edge length different from the one utilized (fig. 3.5). Only at second glance this option reveals its full potential to smoothen our shape:

If we allow for the edges to be of one of two different lengths (say length *A* and length *B*), this results in four different triangles to be combinable (*AAA*, *AAB*, *ABB* and *BBB*). Each triangles implies three inner angles, summing up to twelve angles occurring in this set. Six of these angles are always equal, namely 60° in the equilateral ones. Triangles AAB and ABB are reflectional symmetric, which means they only imply two angles of unique value each. This means that we end up with a total of five unique possible inner angles to utilize in each knot. Thus, we're no longer tied to only one obtainable value for $K(p) = (2\pi - t_p \cdot \pi/3)$ anymore. Rather we can get a much larger set of possible Gaussian curvatures, meaning angle sums by combining different triangles and thereby different inner angles to meet in the node *p*.

To get an idea, of the variety obtainable in *K(p)* we can calculate the amount of different unique inner angles depending on the available edge lengths. First we need to make some definitions:

$\|K(t_p)\|$	cardinality of the set of achievable Gaussian curvatures *K(p)* in *S* at *p*, depending on the number of adjacent triangles t_p (this is the desired variable)
L	a finite set of previously chosen values of edge lengths with $\|L\|$ being its cardinality
T	set of triangles achievable through combination of elements of *L*, cardinality being $\|T\|$
α	set of unique inner angles in *T,* with $\|\alpha\|$ being its cardinality

Furthermore, we have to add an important condition:
Each arbitrarily chosen triple set of the elements of *L* has to be capable of creating a triangle which is *unique* among all the other possible triangles in *L*. This implies that the smallest value in L is larger than 0.5x its largest value (so *any* triplet of values can create a triangle) and that, except for equilateral triangles, no similar triangles (similarity in the geometric sense) occur in the set.

The task is to find the number of unique angles α that occur in these triangles *T*. We will do this in the same way as we did in the previous example of two different edge lengths and six triangles around the node. We simply take the amount of every possible angle that occurs (being the amount of possible triangles multiplied by three) and subtract the ones that repeat. This is on one hand the angle of 60° which occurs thrice in every equilateral triangle adding up to the amount of $3 \cdot |L|$, but we will leave it in the equation once. On the other hand we have to subtract the number of reflectional symmetric triangles multiplied by one (since each of them contains two similar angles, but we only want to count one of them). This is calculated by multiplying the amount of edges $|L|$ with $|L| - 1$, since it is a simple combination of two elements in a set with the condition, that there's no repetition of one and the same value. We can conclude:

$$|\alpha| = 3 \cdot |T| - 3 \cdot |L| + 1 - |L| \cdot (|L| - 1)$$

The amount of every possible triangle $|T|$ derives from the combinatorial multiset:

$$\left(\!\binom{n}{k}\!\right) = \binom{n+k-1}{k}$$

where *n* represents the number of different lengths available and k the number of "picks", meaning the number of sides of a triangle. We don't take the sides order into consideration, since it would only create rotated (and thereby similar) triangles and allow for same length values to repeat. So we will use $|L|$ as n and 3 as *k*,

since we need to combine three edge to one triangle using the variety of $|L|$ different elements.

$$|T| = \left(\left(\begin{matrix}|L|\\3\end{matrix}\right)\right)$$

Now we have everything we need to summarize the desired number of unique angles $|a|$:

$$|\alpha| = 3 \cdot \left(\left(\begin{matrix}|L|\\3\end{matrix}\right)\right) - 3 \cdot |L| + 1 - |L| \cdot (|L| - 1)$$

With these equations at hand we can quickly calculate the number of combinable unique triangles $|T|$ out of the set of edge lengths and the uniquely occurring inner angles in this set. In the next step we would have to examine how $|K(t_p)|$ derives from the variety of inner angles $|\alpha|$. This issue represents a far more complex task and can't be covered in detail within the scope of this work. But we can exemplary calculate $|\alpha|$ with just three different edge lengths $|L| = 3$ and get a variety of $|\alpha| = 16$ inner angles. Using our previous example of six triangles adjacent to the node p, we see, that a combination of six angles out of a set of 16 available ones adds up to a large variety in Gaussian curvature $|K(t_p)|$ obtainable in p. Even though the calculation isn't as simple as that, it's safe to say that this represents a huge increase in achievable smoothness compared to the existence of only one available curvature for each valence of p in the equilateral triangle. Thinking of the actual prefabrication of the pieces, still a set of three different lengths would provide enough simplicity to drastically reduce cost compared to a large amount of unique edge lengths.

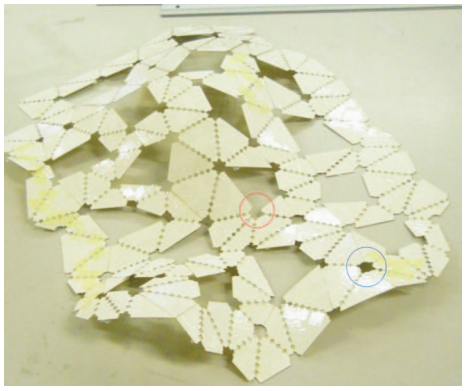

Fig. 3.6: Perforated triangle mesh with limited amount of edge lengths

To enable the structure to fold into its shape without to many constraints, we will still provide some holes in the mesh, mixing rigid nodes to which the earlier examinations apply (blue in fig. 3.6) with completely free edge vertices (red). Experiments with previous models have shown, that a suitable balance between constraint and free nodes is obtained if we follow the rule, that each node must either be an edge node of the red kind itself or be neighbor to at least one these nodes.

This model of a perforated triangle mesh with a limited amount of edge lengths represents the geometric model we will utilize in the further process. To keep things simple, we will simply call it a *p-l-t mesh of the valence x* where x describes the amount of different edge length utilized. Fig. 3.7 shows a p-l-t mesh of the valence 4. In the following, however, it will be of big interest, to actually utilize the theoretical knowledge addressing the potential of high smoothness and flexibility immanent in this structure acquired in this chapter.

3.2 Triangulation

The standardization of pieces in tessellate structures has become an issue of less relevance in the past due to a high efficiency and decreasing cost in the fabrication of structural framework. Against the background of the initial idea of this work to create an easy to use construction kit of agile actuators to be assembled, it is still advisable to adhere to the objective to only utilize a limited amount of edge lengths. This results in a much larger simplicity, especially in the mechanical context. In contrast to architectural applications we can omit the need to achieve a steady appearance of edges aligning in long continuous curves throughout the whole structure. Rather a random tessellation without aligned edges avoids continuous kink folds and thereby stabilizes the structure.
As shown in the previous chapter the triangulation of a given basic surface using a p-l-t mesh won't achieve absolute smooth results. This effect, however, can be minimized by using a larger amount of edge lengths leading to larger complexity or by arranging the triangles on the basic surface S in an optimized way. The following algorithm attempts to do so to make use of the tremendous potential p-l-t meshes provide in smoothness, described in 3.1. It is based on investigations on stigmergy of social insects such as wasps building their nests, described in [8]. The approach is to first subdivide the given surface entirely into triangles of given properties and secondly to remove some of the edges, in order to tear holes in the mesh.

First some definitions:

S	the basic surface. It represents the shape to be approximated.
L	a finite set of previously chosen values for edge lengths with $\|L\|$ being its cardinality
l_i	the elements of L
p	nodes of the polygon mesh to be compiled
b	boundary of the polygon mesh; the closed polyline consisting of all the mesh edges with only one adjacent mesh face (we assume the mesh to be without holes for now, so there is only one boundary polyline)
b_p	normal projection of b on S
R_i	the region of S that is inside b_p
R_o	the region of S that is outside b_p

Algorithm in pseudo code:

1. Set one initial triangle arbitrarily with the condition that all of its three vertices are being placed on S and only l_i are used as edge lengths.
2. Chose one of the triangle's edges and find two new lines (again only l_i) to form a new neighbor triangle on S. This can be done by creating spheres with a radius r_i respective element of L with the adjacent nodes of the chosen edge as their centers. The circular intersection curve c of this two spheres has two intersection points x_1 and x_2 with S, of which the correct one must be chosen as the new node (utilizing the properties of self intersection of new $b_p(x_1)$ or $b_p(x_2)$, or x_i's inclusion in R_i). If c has no intersection point x with S, another triangle edge must be chosen. If there are no intersection points of any c with S, the algorithm stops. Then the solution consists of a polygon mesh with only one triangle. Figures 3.10 and 3.11 illustrate this second step.
3. Update the polygon mesh and b resp. b_p

4. Examine b and pick a single segment or two neighbors of segments of b to operate on. Decide on one of the following options[1]:

   4.a) Repeat step 2 with single segment of b (resulting in the replacement of one segment to b with two new ones, extending b to the outside)

   4.b) If two neighbor segments of b are chosen, fill the narrow gap they span over R_o with two new triangles. This can be done by creating spheres around the three nodes adjacent to the new desired node. One sphere for every l_i with the resp. radius around every node (at a p-l-t mesh of valence 4 this adds up to 12 spheres). These spheres can be intersected, creating a cloud of points (fig. 3.12). Each point naturally has the property of having a distance to all of its neighbor nodes of l_i. From this cloud of points we have to pick the one with the shortest distance to S and with its normal projection on S being outside R_i. This point represents the new node best fitting our purpose, creating the smallest bump achievable in this situation (fig. 3.13). This procedure replaces two segments of b with two new segments outside R_i.

   4.c) If the angle between two neighbor segments over R_o falls below a certain threshold value, directly connect the segments two outer nodes (this replaces two segments of b with one new segment). (fig. 3.14, 3.15)

5. If S is filled with triangles, stop. Otherwise go back to 3.

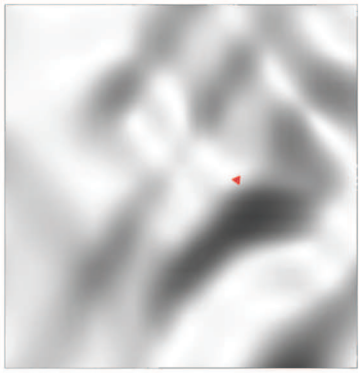

Fig. 3.7: First iteration

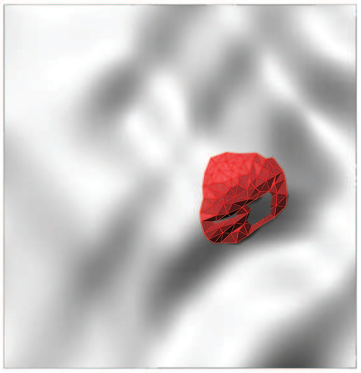

Fig. 3.8: Iteration Nr. 180

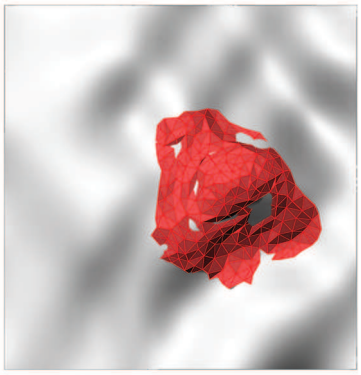

Fig. 3.9: Iteration Nr. 258

[1] The conditions under which the operation mode is chosen represent a tricky task and an issue of optimizing the algorithm's performance. The priority is to find narrow angles over R_o between neighbor segments of b and close them using mode 4.b) preferably or 4.c) in critical situation. The narrowest angle is always treated first. 4.a) is only chosen if neither 4.b) or 4.c) can be applied. These conditions are similar to the behavior wasps show when choosing the most enclosed comb cell to fill when building their nests described in [8]; it is crucial for avoiding self intersections in b_p and make the triangles grow evenly to the outside over S

While the sub method 4.a) produces triangles that are entirely correct in the sense of the task (meaning the edge length is exactly an element of *L* and its nodes *p* lie on *S)*, 4.b) and 4.c) merely attempt to approach this solution. Thus, the algorithm is not capable of producing exact edge lengths *and* placing every node *p* on *S*. One can chose to assign the geometrically inherent error either to incorrect edge lengths, distances of the nodes *p* to *S* or distribute it over both. Figures 3.16, 3.17 and 3.18 show investigations on a sample piece. Graphs on the left show the distribution of the system inherent error from a low variation in edge lengths in 3.17 to a low average distance from p_i to *S* in 3.18. The algorithm was made in Rhinoceros™ and Grasshopper™. Beyond applying the described algorithm I used the physical engine Kangaroo to pull the mesh nodes towards *S* and thereby shifting the error from a bumpy appearance to less exact edge lengths . The work of the algorithm on this sample can be watched in the attached video clip *"04_131107_triangulation_in_progress.mp4"*.

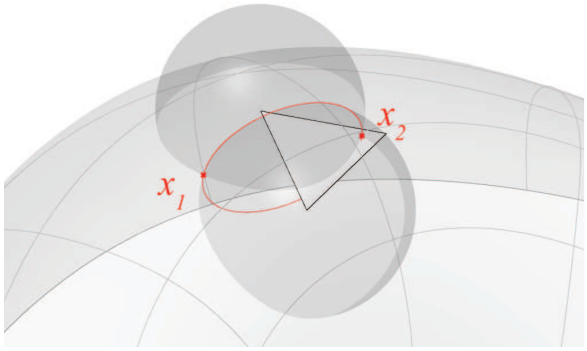

Fig. 3.10: Finding the next node *p* on *S*

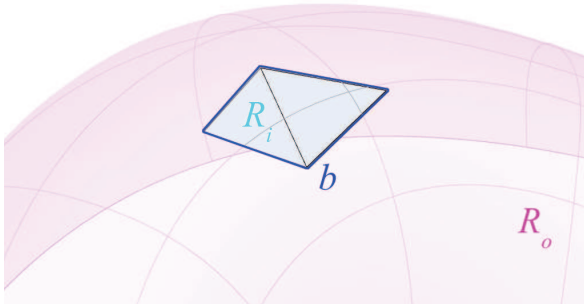

Fig. 3.11: Adding the first neighbor, *b* grows

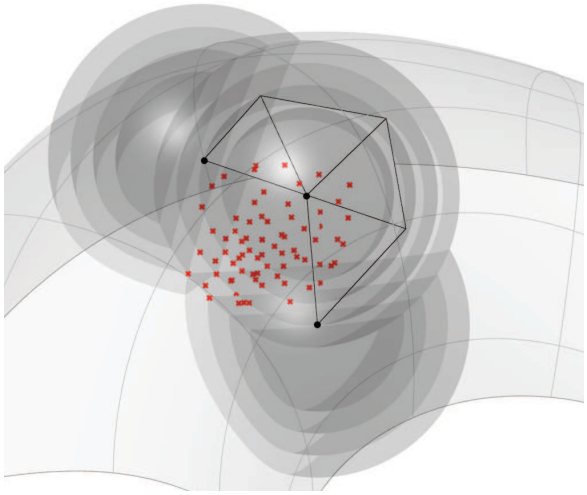

Fig. 3.12: Creating the cloud of possible new nodes

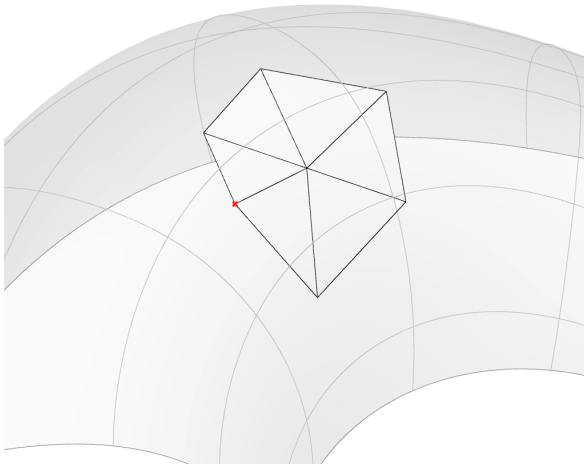

Fig. 3.13: Picking the best fit

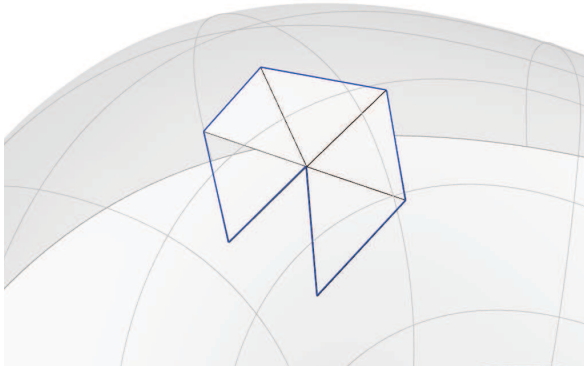

Fig. 3.14: Disadvantageous situation where no l_i fits in the gap

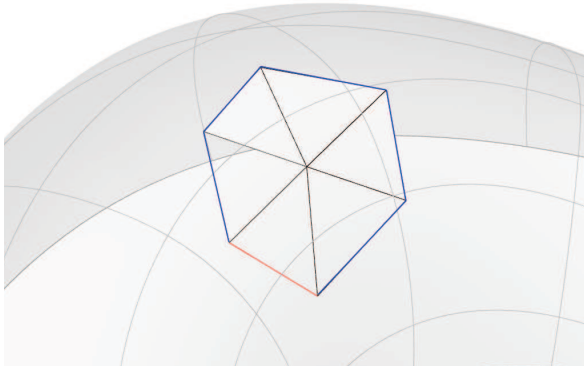

Fig. 3.15: Simply filling the gap

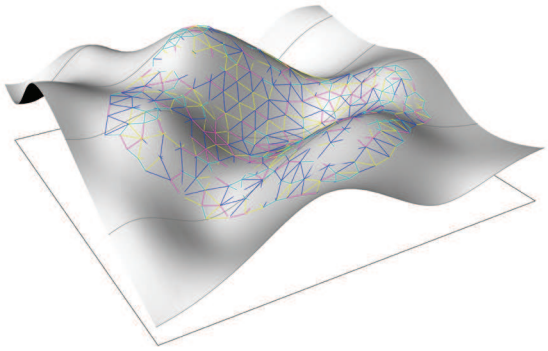

Fig. 3.16: Triangulated sample surface

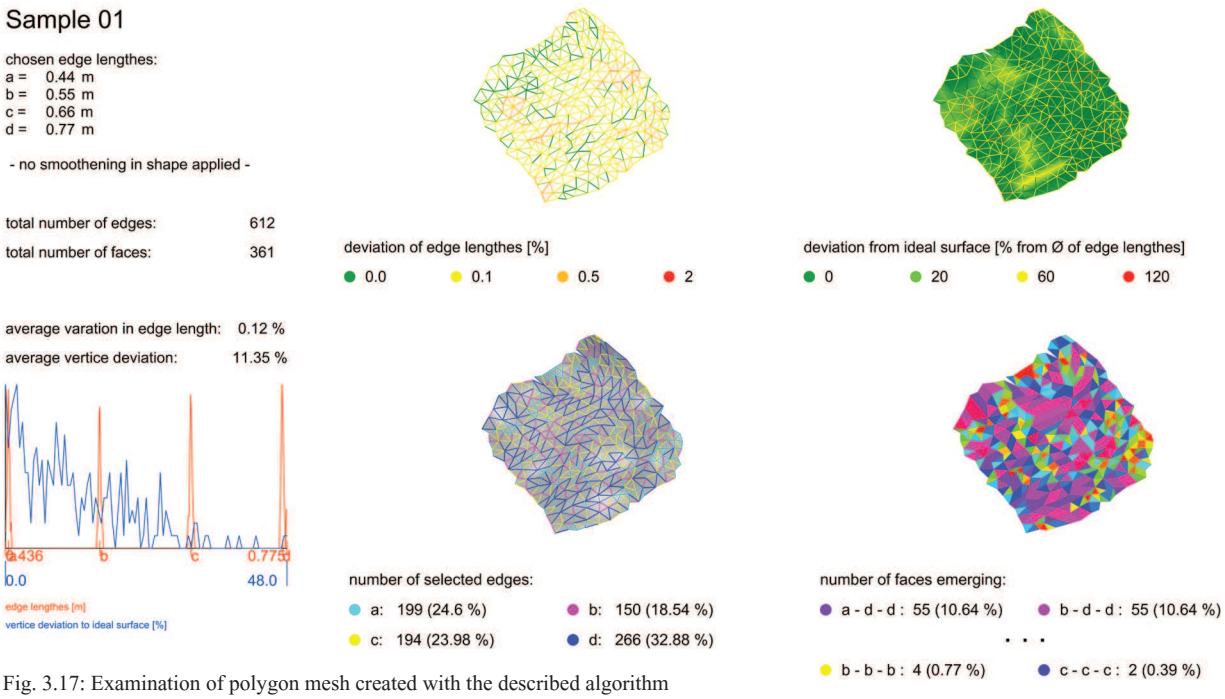

Fig. 3.17: Examination of polygon mesh created with the described algorithm

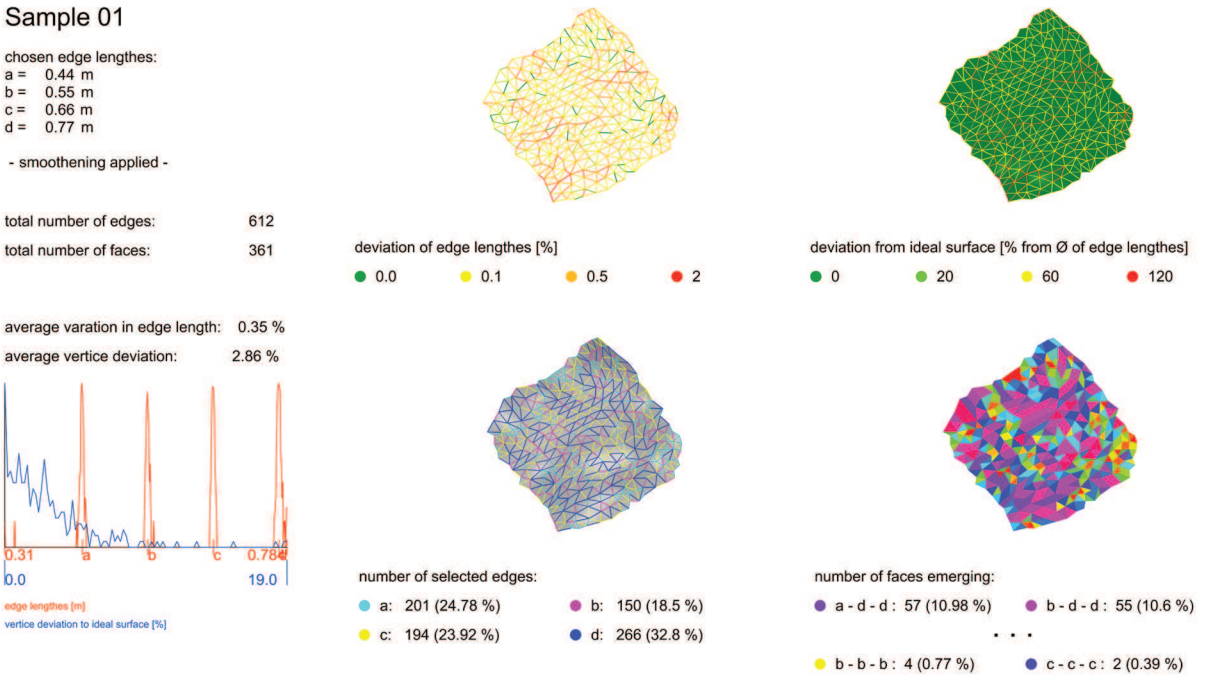

Fig. 3.18: Examination of polygon mesh created with the described algorithm and further smoothening with Kangaroo

3.3 Simulation of folding kinematics

The software of choice to simulate the kinematic behavior of our folded structure is Rhinoceros™ along with the plugin Grasshopper™. Furthermore the physical engine Kangaroo was used. By simulating a network of interconnected springs, Kangaroo able to find an equilibrium of a large number of interdependent forces represented by this set of interconnected vertices. Although it was actually designed for this purpose we can still make use of it by simulating very stiff springs and thereby turning them into rigid edges without the ability to change their lengths. This working mode, however, actually does not exactly suite the task.

A geometrically correct approach on the other hand was introduced by Tomohiro Tachi in [9]. Based on recent proceedings in origami mathematics [10] he elaborated a way to implement the present theoretical knowledge of the mathematics behind origami folds into a Java script and thereby simulate folding behavior of rigid structures in the mathematically correct sense. His approach is based on the fact that in an enclosed node, without adjacent holes, the rotation of one piece *t* causes the other pieces adjacent to the same node to rotate about their individual rotation axis in a way, that the product of their rotation matrices equals the identity matrix *I*. This fact was shown in [10].
It can be expressed as:

$$F(\rho_1, \ldots, \rho_n) = \chi_1 \cdots \chi_{n-1} \chi_n = I$$

Where χ_1 to χ_n are the individual rotation matrices of the adjacent facets. The key of simulating rigid folding behavior correctly lies in dynamically solving this equation in every time step of the simulation and combining many nodes in an equation system. To do so, Hirotachi uses a conjugate gradient solver, but also a Broyden–Fletcher–Goldfarb–Shanno solver would be thinkable. The implementation, however, of Hirotachi's method into a plugin for Grasshopper written in C# would represent an adequate translation for our purpose. Unfortunately, this could not be done within the scope of this work.

However, since we utilize the previously described p-l-t mesh which combines properties of both, encompassed and free edge nodes, we have a relatively large amount of nodes (namely the free ones), Hirotachi's approach doesn't apply to. The reason is that the very basic condition his algorithms are based on, that the sum of adjacent angles around a node is fixed, doesn't apply to partly open nodes.

In addition, examinations with the Kangaroo plugin showed, that, applied to a p-l-t model only small variations in edge length occur (ranging in an average dilatation of about 0.1%, when super stiff springs are simulated). Furthermore, peaks in length dilatation can be measured and interpreted as a graduator for constraints and jams occurring in the process of folding. We will discuss this issue a little later.

Let's first a have look a the data structure of a triangle mesh as it is used in computer graphics:

The data, a commonly used face-vertex mesh comprises consists of a list of vertices each with its position and a unique index. Faces can also be

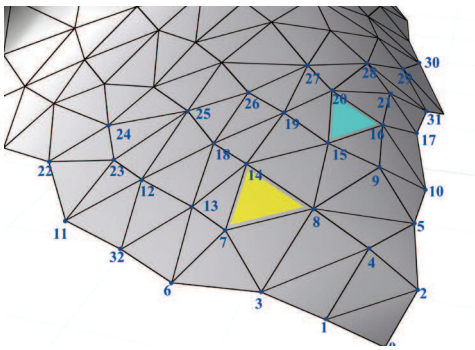

Fig. 3.19: data structure of a face-vertex mesh

arranged in a list and consist of three integers, representing the index of the vertex they are adjacent to. In figure 3.19 for example, the yellow face would consist of the integers 7, 8, 11. When the positions of the vertices change, their indices and thereby the face pointing to these indices remain invariant. Thereby the list of faces represents a connectivity map that is independent from the actual position of vertices. The order in which the faces corner vertices are noted define their orientation (e.g. counting vertices clock-wise means that the normal vector points “up”, counter clock-wise means “down”, respectively “up” and “down” can be replaced by defining the direction “inside” or “outside” to depict a volume).
Due to advances in work flow and higher accuracy in identifying and naming edges in the dynamically changing test setup we will use a different kind of mesh representation: the half edge data structure. This system also implies a list of vertices as a base. The connectivity map, however, consists of a list of vertex indices representing the edges. Every edge occurs twice respective to their starting vertex [11].

To prevent the vertex indices from mixing over time, which would result in an adverse change of the connectivity map of our triangle mesh in every time step, we *first* set an all time valid connectivity map using the half-edge mesh structure. Note that this is exactly what the triangulation algorithm in 3.2 is doing (besides it also defines vertex positions) This forms the very base of accessing single edges and reading their relevant values (angle, edge length dilatation etc.) while their positions change dynamically. Dynamically means, that in every time step, we get a completely new set of vertex positions. Their connectivity map, however, remains invariant.

Using this structure various simulations were run. The video clip *“05_131118_collapse_re-erection”* in the appendix shows the collapse of a folded p-l-t mesh of valence 4 and its reerection. This reerection was achieved by simultaneously applying a hinge force on every two adjacent facets defined by their common edge (in the following we will just speak of the “angle over an edge”). The desired angle this force is performing towards is the same angle, which was measured in the very beginning, before the structure collapsed. So this simulations represents an exact image of the behavior we intend to obtain:

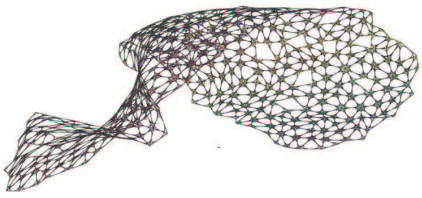

Fig. 3.20: Simulation start; ideal situation

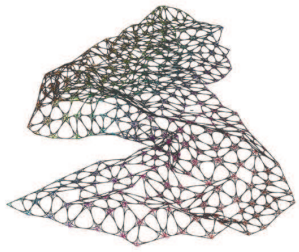

Fig. 3.21: Moment of total collapse

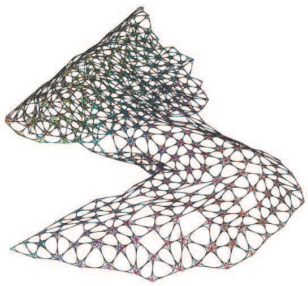

Fig. 3.22: Moment of force initiation

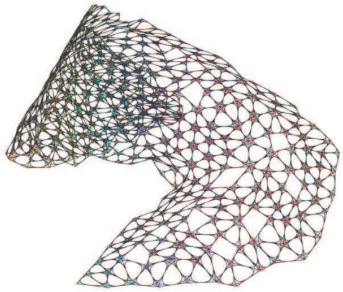

Fig. 3.23: Complete re-erection

In the beginning the structure is complete (fig. 3.20). It consists of a p-l-t mesh of valence 4 approximating a shape we defined as the target shape. The information of connectivity, the vertices exact position and the specific angle over each edge (actuator) in this ideal situation is known and stored. Then the simulation starts. By applying gravity but no resistant force, the system collapses entirely losing the information of the vertices' resting position in the target shape. In the moment of total collapse and no movement captured in fig. 3.23, the only information the system has is the connectivity of its particles and the target angle each edge has to have in order to represent the desired shape. In fig. 3.22 the hinge force pressing each edge back into their target angle just started. After a short time the structure is re-erected back into its initial, desired

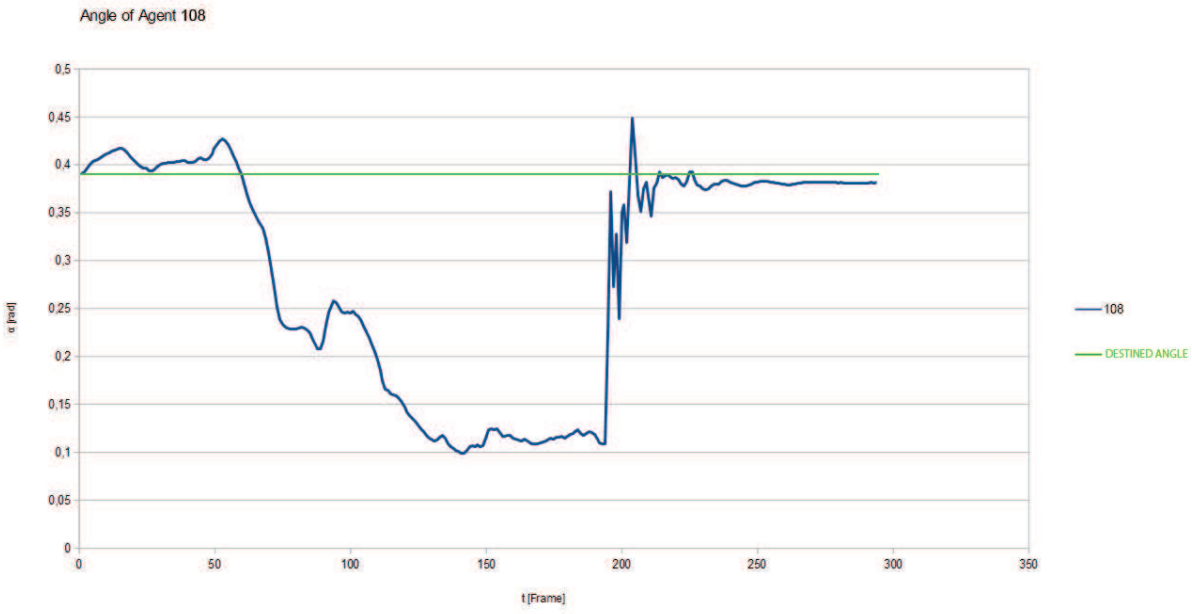

Fig. 3.24: Angle curve of agent nr. 108

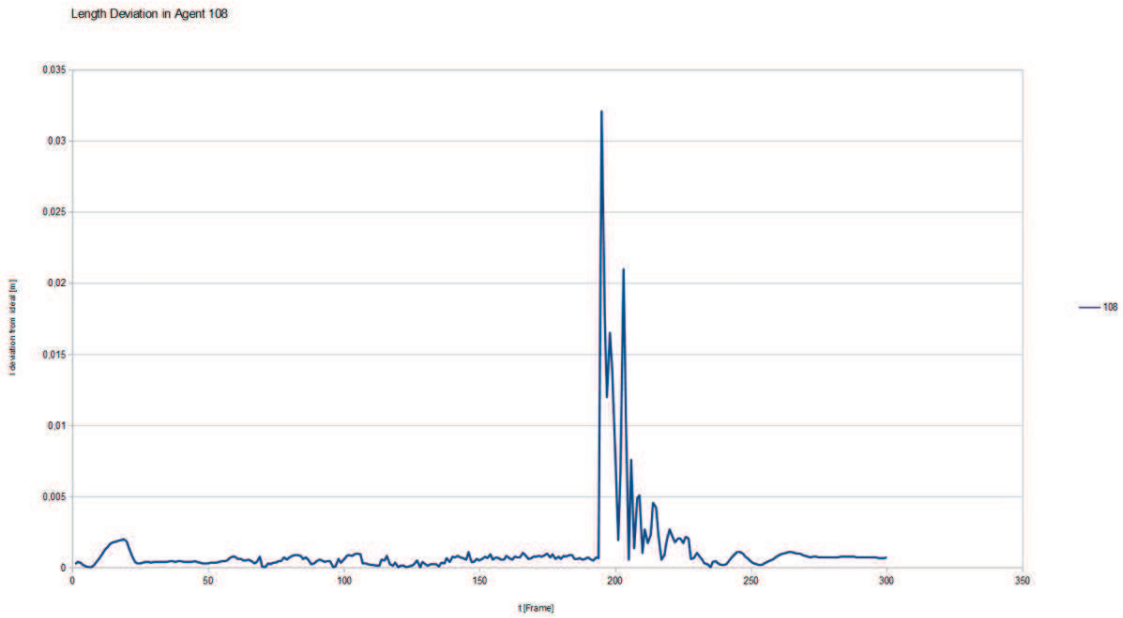

Fig. 3.25: Length dilatation of edge nr. 108

shape. Fig. 3.24 shows the angle curve of an exemplary edge (nr. 108 to be specific) over the time of simulation. The moment of the hinge force's application can clearly be seen in the upward kink.

This simulation shows us, that the information of the connectivity and the target angles over each edge suffices to definitely regenerate a desired shape. The vertices again take their desired position. But it also discloses another issue:
Fig. 3.27 shows the deviation of the edge length of agent nr. 108. In the moment of the hinge force's application the length deviation rises abruptly. Due to the way the physical engine of this simulation works, by simulating elastic springs between particles, the structure remains connected. But in a real world application, the edge would simply be torn apart at this moment. From this follows, that, though the information of connectivity and target angle is enough to obtain the target shape, it is not enough to simply "switch on" the agents in order to smoothly fold into this position. This action would cause the structure to crack. We need to enable the edge actuators to smartly coordinate their behavior.
One option to achieve this would be, to elaborate a sophisticated simulation, which takes the system inherent constraints into account and record the angle curve for each actuator, in which they don't cumber each other. We could also simply revert the angle curves of the collapse, since in this process the system , since it represents the transitions from one shape of the system to the desired one, without cracking the actuator. This sort of pre-recorded "choreography" of harmonized control curves of the actuators could be applied to each actuator in the specific connectivity map with a very specific starting position. Every slight breach in individual pieces would cause jams in folding with unforeseeable consequences for the system. We will still keep it in mind as a very simple first step, a "demo" of our desired behavior. Due to the initial idea to create a flexible construction kit, which can be reassembled in order to create different shapes, however, this operation mode doesn't seem advisable.

3.4 Simulating the collaborative behavior

Here the idea of a collaborative system again comes into play. What if, instead of specifically controlling the movement of each actuator, predetermining their control curve, we give them a strategy to independently find their way towards the desired solution? And thereby giving them the tool to adapt to *any* desired shape, if we just change the value of the desired individual angle? The mode of operation would remain the same: Reach you desired angle! Don't crack!
To do so, the agents would have to have knowledge about their target, the present angular state and a set of rules on how to operate in reaction to this state in order to find the predetermined value of their ideal angle. Furthermore the agents need to be interconnected in order to exchange information about their states to avoid jams and dynamically adjust their control curves among each other. (fig 3.28)

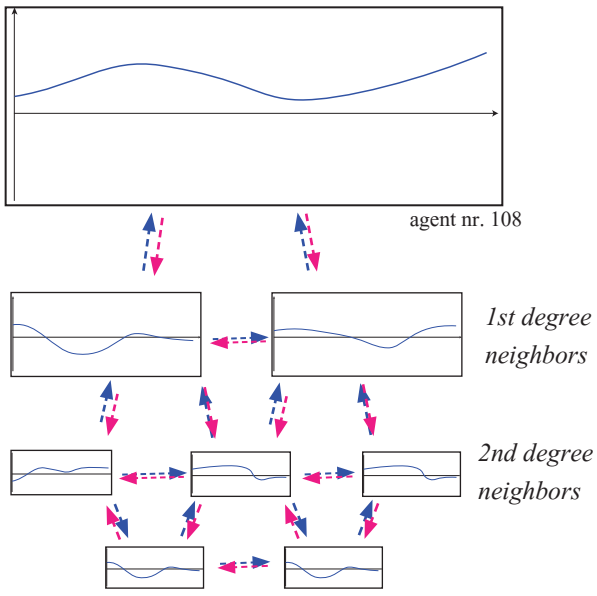

Fig. 3.26: Scheme of interconnected control

The task derived from these aspects is to find the correct set of rules the agents need to utilize. More specific, rules about the conditions under which communication signals would be sent and how they would be interpreted and reacted to, once they were received.

To do so, several investigations on how signals would spread over a p-l-t mesh's connectivity map were done, leading to different models of signal conductions depending on whether or not they would be erased when interfering with one another.
The video clips *"06_131106_degrees_of_neighborhood.mp4"* and *"07_131106_simple_vs_self_intersecting_signal_spread.mp4"* illustrate these categorizations. After dealing with this task for quite a while, it became obvious, that a simulation of this complex behavior would cause more problems than it solves. If it should work absolutely correct and depict the way a real set of actuators collaborates dynamically, it would become a time consuming task to establish the respective algorithms. In this context, the question arose, if a physical model, a prototype would provide a more feasible solution.
An experiment had to be done.

4. Physical modeling

Though different simulations were executed and optimized to get an idea of the physical behavior of our system, their actual validity remained speculation. More importantly, the simulation of signal conduction and reaction proved quite complicated. Thus, a physical prototype had to be established. Why is that?
The question to investigate on remained the same: Which set of rules of interaction does a simple agent, like an angle actuator on a p-l-t mesh need, to collaboratively reach the goal of achieving a certain state (position and angle), it wouldn't be able to reach on its own? Can the system of interconnected entities, furthermore, even reach a sort of collective intelligence consisting of nothing else than the sum of its individual actions and the exchange of its signals? How would this communication have to look like? The question remained the same, only the approach towards the problem changed: Since our search for these set of suitable rules, for the design of the actuators *mind* proved highly complicated if we approach it through pure simulation, in other words, through pure thought, without an actual materialization, we wondered: Would a physical model, a prototype, simplify this search? Would it even disclose possibilities we were unable to see before?
This question, however, isn't new. In Jörg Petruschat's and Julian Adenauer's opinion, the answer is clear: In [12] they take a stand for the inseparability of the idea of a prototype's functionality and the material it consists of. They even go a step further by stating that in *prototyping* lies the only operation mode capable to innovate in s design context, since the informational feedback a prototype gives enables us to enhance its performance. Even research through prototyping in a design context is a valid topic; the idea to gain knowledge through designing.
The other aspect is that of artificial intelligence which was broadly discussed by Marvin Minsky. In [13] he investigated on the question, what the human mind consists of and how the blurry, unseizable haze we call consciousness, can be structured, partitioned and rebuild in machines. Three of his very fundamental assumptions are, that firstly the mind can only emerge from a body, secondly consciousness and thereby intelligence is composed of a very distinguishable and quite understandable set of interdependent components and clusters of components and third, that our unawareness of these simple components derive from their sheer complexity in interdependence and manifoldness. Our inability to comprehend them leads to the believe of an outer body mind, some immaterial spirit that steers our physical materialization. Minsky's ideas were recently backed up by modern research in cognitive sciences. In [14] Antonio Dimasio locates the interface between the human body and mind in two central cores of our brain: *nucleus tractus solitarius* and *nucleus parabrachialis* aren't only the structures that constantly receive feedback from the body. Signals of pain, cold or hunger arrive here. They are, scientifically verified, also the structures where basic emotions like joy or anger arise. Dimasio further argues, that these very core emotions form the basis of consciousness and thereby intelligence. Connecting this to the ideas of Minsky, one can state, that our consciousness is nothing but a set of simple components, reacting to one another and that the very fundamental components, the main triggers directly derive from the physical feedback of our body; furthermore this system can be imitated in machines. These arguments militated in favor for our approach. We had good reasons to be optimistic.

4.1 Test Setup

The experiment had to serve both: a sufficiently large number of actuators to allow for enough complexity in interaction, but also affordability and feasibility in a relative short period of time. Furthermore it needed to provide enough flexibility for the system to move relatively free but also to be stable. For this purpose a p-l-t mesh of valence 3, consisting of 22 inner edges (actuators), 19 outer edges, 21 faces and 24 vertices was established. It consists of both, encompassed and edge nodes. A large hole in the middle serves as the necessary perforation to provide flexibility. The mesh layout approximates a smooth shape defined previously (fig. 4.1). We could imagine this initial free form surface to be a pavilion or a garage for instance, the discretized structure (fig. 4.2) as its formwork and the starting state (fig. 4.3) as the formwork's pieces assembled on the floor to be folded into shape. The final surface was triangulated using the tools described in 3.2 (fig. 4.2). It contacts its surroundings in three points: on the ground and two anchor points of higher elevation.

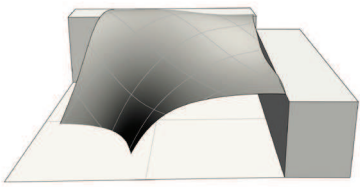

Fig. 4.1: Target surface

Fig. 4.2: Triangulation

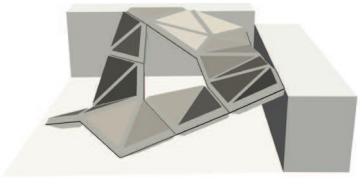

Fig. 4.3: Transformation through folding

The three different edge lengths where chosen to be 0.28m, 0.34m and 0.40m. Due to our analysis in 3.1, we would receive 10 different possible triangle configurations, derived from the edges combinatorial multiset. In respect to the built height of our agents, giving facets a definite up side and down side, however, we have to count triangle ABC's reflective counterpart BAC as well, resulting in a set of 11 possible combinations (fig. 4.4). Not every configuration is present in our example. Thinking of a possible utilization of a flexible concrete formwork the setup size would be commensurate with a scale of about 1:5. The informational system it seeks to investigate on, however, is without scale.
This arrangement was taken as a basis for the actuator's further design (motor size, choice of hinges and so on). An exemplary simulation of its movement can be seen in the video clip *"08_131219_demo.mp4"*. This simulation also formed the source of control curves utilized later.

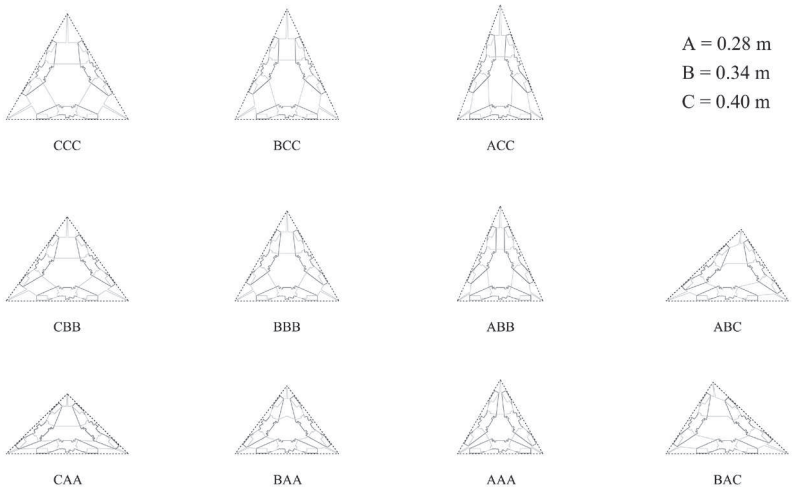

Fig. 4.4: Set of possible triangle configurations

4.2 Mechanical Basis

In order to quickly make advances towards an adequate setup in mechanic and electronic terms, smaller fragments consisting of a maximum of two or three actuators were established, evaluated and redone. Once their virtue was proven, they were meant to be produced in large amounts. This resulted in a total of three (and a half) prototypes which will be described here. But first we'll have a look at the very basis of the functioning.

Researching about the different kinds of motors and mechanic parts available on the market, two basic ways to empower the actuators arose:
The first one consists of a slow moving, relatively precise servo motor as it is commonly used in model building (fig. 4.5). The motor could have been placed in a way that its rotation axis aligns with the edge between two mesh faces, so that the cogs of its wheel could directly manipulate the position of the attached triangles. A big advantage of these servo motors is, that one is able to directly control their angle value. With the help of an implemented potentiometer and gears the motor slowly adjusts towards a prompted position. Upon further testing and trying, it was figured out, that the force the servo motor is capable of applying is quite low compared to their size and weight. Furthermore a motor of this kind capable of lifting the estimated loads would have been quite costly.

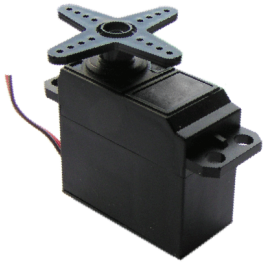

Fig. 4.5: Servo Motor [f]

The most promising alternative was represented by the idea of connecting a small, weak, but fast spinning DC motor (fig. 4.6) as it used in RC cars and model crafts to some kind of threaded shaft. Through rotation the motor would be capable of strongly pushing or pulling a counter piece of the shafts thread (e.g. a nut) towards or away from itself. Along with a certain built height on each triangle, this pushing and pulling could be transformed into a rotation of the triangles around their common edge. To do so, the planar triangular faces had to be extruded along their normal direction, establishing a frame to carry the empowering parts and turning push and pull into rotation. Thus, the flexibility range in which a triangle could rotate around its common edge with a neighbor triangle would be limited due to contingent intersections of these built up frames. Figure 4.7 shows the basic principle and the desired acting field of the actuators.

Fig. 4.6: DC Motor [g]

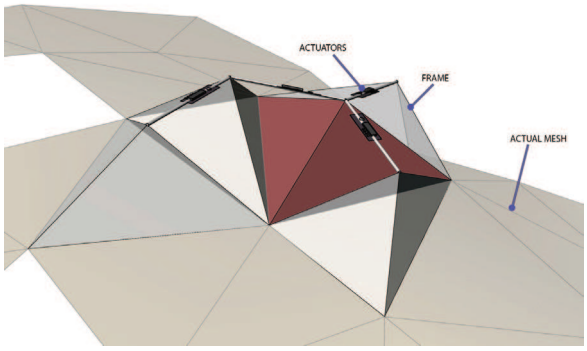

Fig. 4.7: Frames on the mesh

Further sketching and testing resulted in a continuous, stiff frame based on the planar base of each triangle. In order to flexibly connect it to its neighbors, it has to provide some sort of contact area on the triangles' outer edge (fig. 4.8 in orange). The forces of attraction / repulsion between neighbour

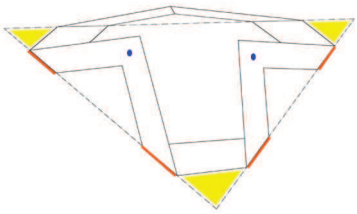

Fig. 4.8: Basic layout

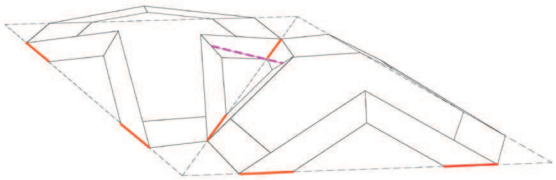

Fig. 4.9: Two neighbors connected

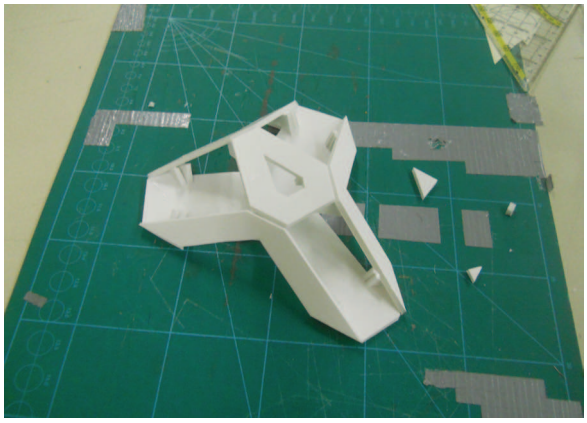

Fig. 4.10: Mock-up

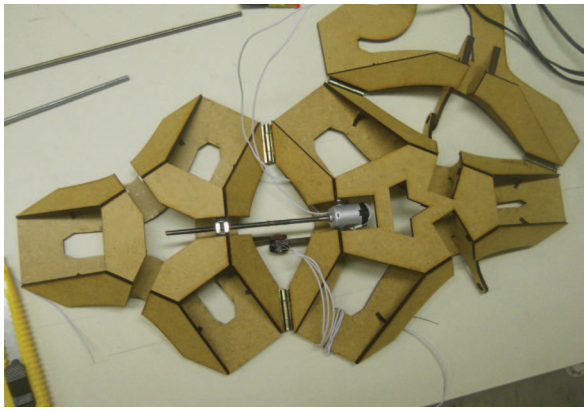

Fig. 4.11: Kingfisher 0.1

frames has to be applied in the area of the blue point. The force axis can be seen in magenta in fig. 4.9. Besides, the areas near the triangles' corners (yellow) had to be kept free to avoid deadlocks and unwanted constraints in nodes with many adjacent facets. To assure the shaft to move freely the area below had to kept open as well. Otherwise it could intersect with the triangles base plate when its relative angle exceeds a certain value below zero.
This basic ideas were built in very simple mock-ups out of foam cardboard as seen in figure 4.10.

With this very basic idea of a potential operation mode, it was possible to construct the initial prototype. I named it "Kingfisher" and made the crown its emblem. (fig. 4.11)

4.3 Kingfisher 0.1

The motor utilized in this model was much of a random pick made under the conditions of a small size and an operation with 12V. It is a Motraxx X-Slot Race 143 with an idle speed of about 22000 rpm and (more important) a torque of 1.3 Nmm (manufacturer information). The testing of this prototype with an Arduino™ Duemilenove can be seen in the video clip *"09_131215_KF01.mp4"* in the appendix. The major insight this model gave was that the angle between the triangle's base plate and the built up frame along with the motors pure power is of crucial importance to the actuators angular operation range. This aspect shed more light on the correlation between the section of the frame and the prototype's maneuverability. To discuss this issue, we first need to make some definitions.

Figure 4.12 and 4.13 show the basic variables, that most significantly affect the moving behavior. For further understanding following variables are of importance:

A	the center point of rotation, being the middle point of the common edge to rotate about
B	the rotation axis around which the motor and thereby the force applying shaft rotate, relative to the frame
B'	the equivalent counter piece to B on the opposite site
s	the axis of the force applying shaft, the connection between B and B'
a, a'	route connecting A and B, resp. A and B'
α, α'	frame angle between the base and *a*, resp. *a'*
c	the circular track which *B* and *B'* describe on their rotation about *A*
β	angle between the shaft's axis *s* and the tangent of *c* in *B'*

It is important to remark, that the frame doesn't exactly align with these major axes due to thickness of the material and necessary displacements. This is why we'll define (fig. 4.14):

α_c, α_c'	angles between the base and the frame's front shield
d, d'	clearance distance between the motors' resting axis and the frame
h	built height of the frame

Even though the different suspensions of motor and counter piece require different constructions and thereby different clearance distances in both sides, we'll simplify the construction by planning the frames side symmetric.
Figures 4.16 and 4.17 show the section of the frame when a rotation is applied, with a certain angular range, the actuator is capable of accessing. The range's lower resp. upper bounds r^- and r^+ define the actuators field of action. Figure 4.17 also shows, that the tip of the shaft *s* gets close to the triangles base plate in higher ranges of r^+. Since the shaft might penetrate the base plate when the rotation *r* exceeds a certain value, this represents an issue we'll have to take into account when increas-

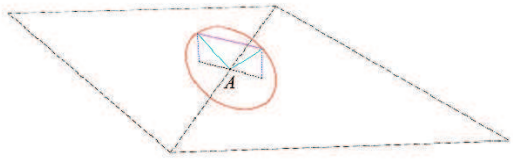

Fig. 4.12: Basic geometry - perspective

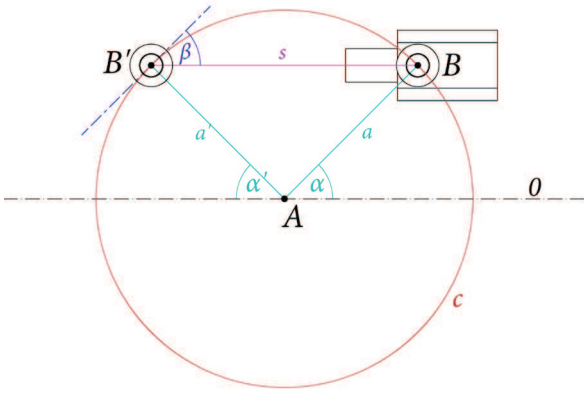

Fig. 4.13: Basic definitions in section

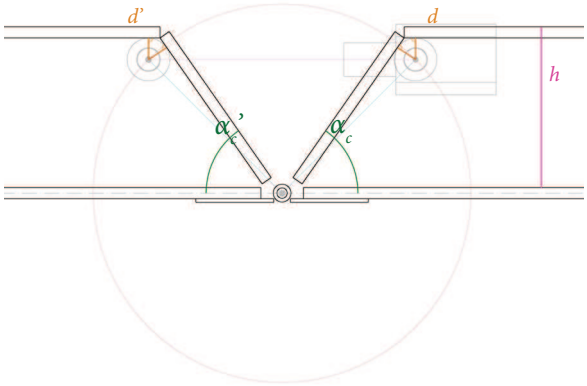

Fig. 4.14: Frame section

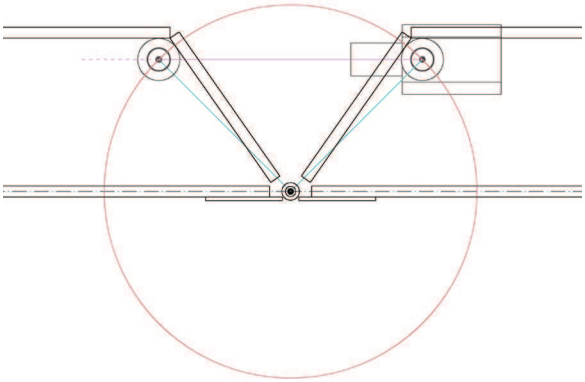

Fig. 4.15: Initial situation

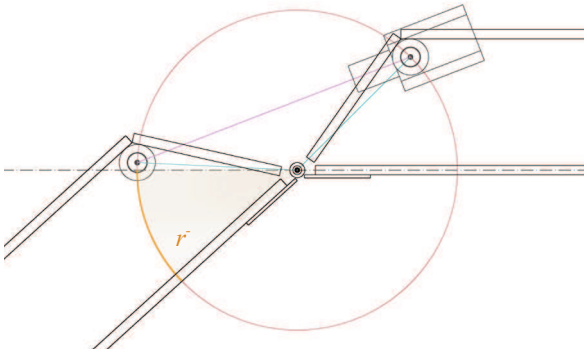

Fig. 4.16: Rotation below 0°

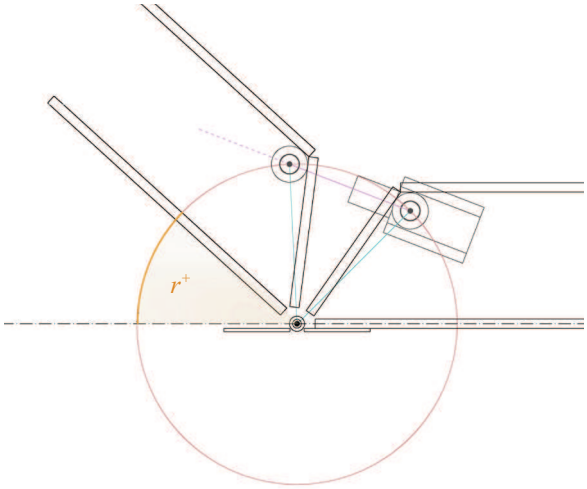

Fig. 4.17: Rotation above 0°

Fig. 4.18: The lower bound of Kingfishers' operational range r^-

Fig. 4.19: The upper bound of Kingfishers' operational range r^+

ing the flexibility of our actuator.

With these definitions we have the necessary tools to definitely describe Kingfishers' capabilities. The time frame 0:13 min to 0:16 min in video clip "*09_131215_KF01.mp4*" shows that this prototype was not able to lift itself off the ground by pushing its neighbor into an angle below zero degrees. Also it could not pull its counter piece above an angle of about 65° (figures 7.14 and 7.15). The latter being induced by inaccurate assembly of the potentiometer in the middle, this prototype was able to pull together until both the front shields of both sides would meet.

Since the model was constructed with an angle α_c of 45° resulting in an angle α of 35° we can conclude, that the first prototype had the ability to operate on a range r^- to r^+ of 0° to 90°. The model also showed, that the upper bound r^+ is directly defined by α_c (in a symmetric alignment r^+ would be 180° - 2 · α_c) and r^- follows from the working angle β: In low angles, the applied repulsive force in the shaft is more and more transformed into a force pushing the pieces apart, rather then making them rotate about their common edge. Apparently the working angle β equals 90° - α = 55° in the position of r = 0°, so with $\sin(55°)^2$ being 0.329 it can roughly be estimated that only about 30% of the motor's force was actually turned into a rotation, the rest was lost to the act of pushing the pieces apart. Beside the relative heavy material utilized (MDF), the weak motor Motraxx X-Slot Race 143 and the inaccuracies in assembling the model, apparently the section and especially the frame angle α_c had to be optimized in the next prototype in order to achieve a higher operational range and (more importantly) to distribute this range evenly in a way that $r^- = -r^+$.

4.4 Kingfisher 0.2

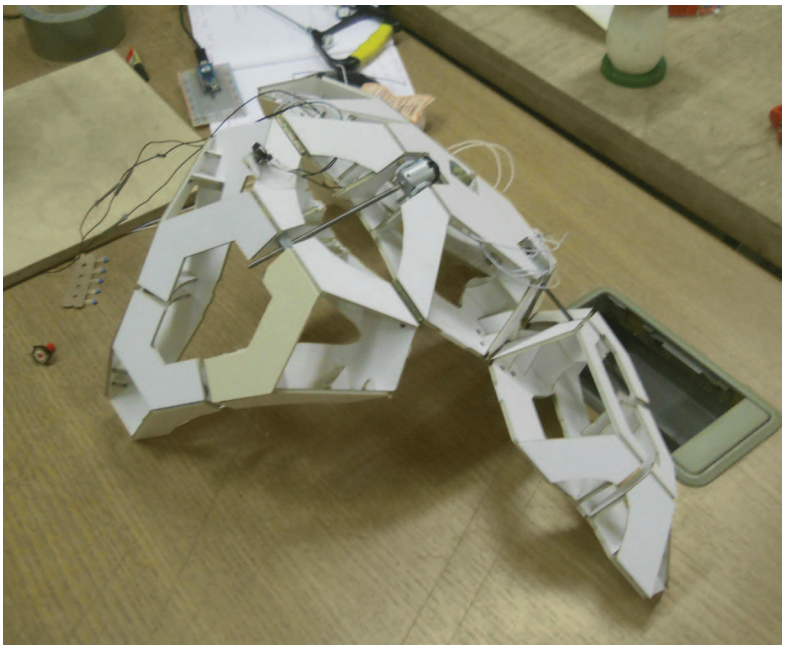

Fig. 4.20: Kingfisher 0.2

As a conclusion from the previous prototype it was estimated, that, using the mechanism at hand, the actuator is capable to effectively function up to a working angle β of about 55° and the operational range is about 90°. Now the frame angle α_c had to be changed in order to distribute this range evenly from $r^- = -45°$ to $r^+ = 45°$. Slight modifications in the way the suspension in *B'* was designed led to an increase of the clearance distance *d'*. Furthermore the built height h was raised from 3.65cm in the first prototype to 5 cm to increase leverage about A. With these modifications, the given material thickness and the hinge design a simulated annealing solver based the on this geometric setup could be applied. The suitable value for α_c was found at 56.7° resulting in a (theoretical) operational range of -43.4° to 43.4°. Kingfisher 0.2 was designed using these parameters (fig. 7.12) and a much lighter foam cardboard material. Besides, different kinds of stronger motors were tested. This increase in power wasn't taken into consideration in the design of the model's section, since much higher stress had to be assumed, when a higher number of actuators is combined (larger weight to lift). Furthermore the previous prototype acted quite

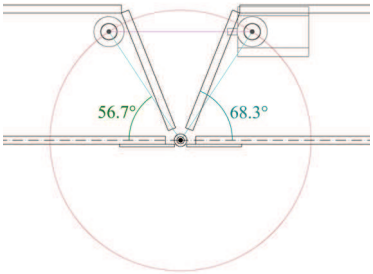

Fig. 4.21: Section of Kingfisher 0.2

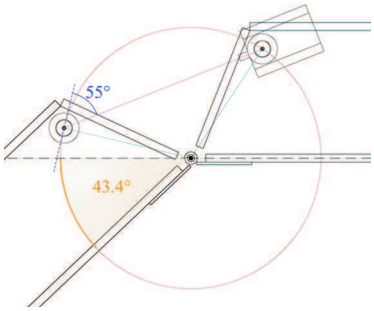

Fig. 4.22: Section of Kingfisher 0.2 in r^-

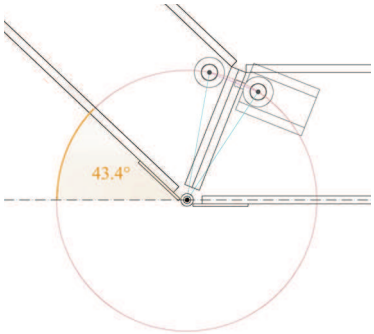

Fig. 4.23: Section of Kingfisher 0.2 in r^+

weak in the area around $r = 0°$. The gain in the motor's torque was to be fully set off these aggravations. The surmised improvement in its maneuverability was confirmed and even outstripped in the testing with different motors. As figure 4.20 and the clip *"10_131222_KF02.mp4"* in the appendix show, Kingfisher 0.2 can easily lift itself of the ground and overshoot the calculated lower range bound r^-. It also moved much quicker than its predecessor. Still the actual loads to be lifted once a large amount of actuators was combined was hard to appraise. However, the moving speed of this prototype was also a disadvantage since it made it harder to react to unforeseen interruptions. The ideal motorization was envisioned as quite steady, yet powerful. This was the reason to consider a gear box in the next version.

In the final prototype, many improvements were to be realized. In order to get a steady, yet strong impellent, a gear box with a ratio of 1:4 was introduced (fig. 4.24 and 4.25). Therefore the shaft now rests on bearings (blue in fig. 4.25) and the motor (red) is turned around, showing the front its rear end. Further, in the Johnson 20543, a powerful (13Ncm torque), slow (maximum of 7000rpm) yet very affordable (only 1,45€) engine was found. It's higher weight of 36 grams compared to the previous models tested was estimated to be set off its higher torque. With the force's multiplication by four through the gearbox, one could confidently increase the range r^- to r^+ in which the actuator was supposed to operate by lowering α_c and assume, that the power assigned was high enough, to lift more than one piece, within the whole domain. Furthermore a more firm mounting for the potentiometer was implemented, to prevent the system from constraints in the potentiometers shaft jamming and blocking the rotation due to imprecision of its fixation. In addition, as against previous models, the actual actuator was separated from the triangles carrying them. Doing so, a uniform actuator with a standardized bore layout, applicable on any triangle baseplate with the required bore hole layout was established. The challenge in this step was to find a shape, that works with every triangular layout. Furthermore potential penetrations of the threaded shaft with the facets base had to be avoided in any triangular configuration possible. This is the reason why the triangle baseplate was gradually degraded to a few connector pieces in the further design process. Also it led to some complications in the actuators design on the non motorized side, resulting in the implication of an aluminum spine to achieve a much thinner, yet stable basis.

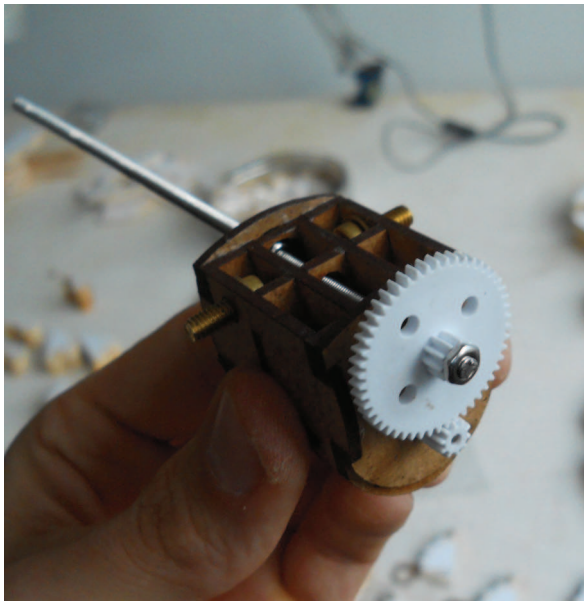

Fig. 4.24: Gear box

Besides, a lighter plastic hinge was used instead of the heavy, imprecise metal hinges used before.
The approach towards its construction, was to take the ready-made, inalterable pieces like the plastic hinges, nuts, screws, the motor, the gear-wheels, the shaft and the newly implied aluminum spine as the origin and then establish a frame carrying them out of MDF and PU foam cardboard sheet. Parts which were set to bear higher loads, and still keep their shape precisely were made out of MDF, the rest, to save weight, was planned in foam sheet.
To elaborate the proper section of the frame the same method as described in 4.4 was used. This

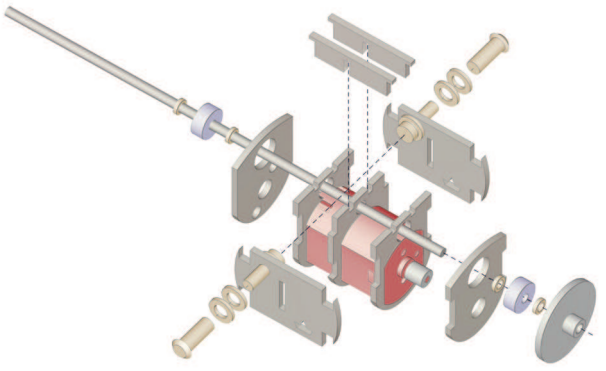

Fig. 4.25: Gear box, exploded

time the constraints were a bit different:
A feasible working angle β was estimated to be 75° instead of 55° degrees assumed before. This alteration derived from several considerations:
The motor used in the previous version was a Motraxx X-Slot 10S Race with a torque of 8Nmm. Our new motor, the Johnson 20543 came with a torque of 13,49Nmm (according to manufacturer information). In addition, a translation of 1:4 was applied and increased the actual torque to 54Nmm. This represents a power 6.8 times higher than in the previous version. To take the higher friction in the gears and the increased weight of the new motor into consideration, this factor was reduced to four. With this estimation of the power increase a simple calculation shows, that we can increase β to 81.8° if we want to achieve the same force acting in tangent direction as in the previous example with $\beta = 55°$: $\cos(81.8°) = 4 \cdot \cos(55°)$
In these areas of β, however, another important aspect comes into play: With the working angle β being 81.8° the force acting in radial direction is about 6.9 times higher than the force acting in tangential direction. Since we built the model and especially the frame to be very light, the possibility to crack the frame with this force,in the case the rotation is blocked, has to be taken into account. To avoid that risk, β was lowered to 75°, where the factor between tangent and radial force is only about 3.7. Doing this, we cut almost half of the stress applied to the frame by just slightly reducing the actuators flexibility. A justifiable step. The rest of the section (fig. 4.26, 4.27, 4.28) was designed around this basic operating angle of $\beta = 75°$, the clearances *d, d'* derived from the pieces used to mount the gearbox and the shaft's counter pieces to the frame and some other parameters. These requirements resulted in a rotational range of $r^- = -62.5°$ to $r^+ = 62.5°$ (fig. 4.27 and 4.28).
Figure 4.28 clearly shows, that the shaft (magenta) pierces the base plate. In respect to this issue it made sense to further investigate on the shafts movement. Video clip "*11_131229_shaft_trail.mp4*" in the appendix shows an animation of how

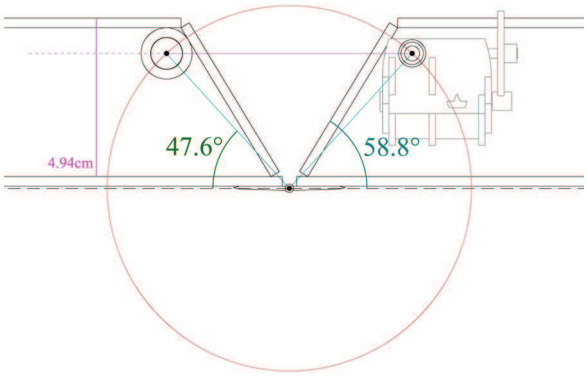

Fig. 4.26: Section of Kingfisher 1

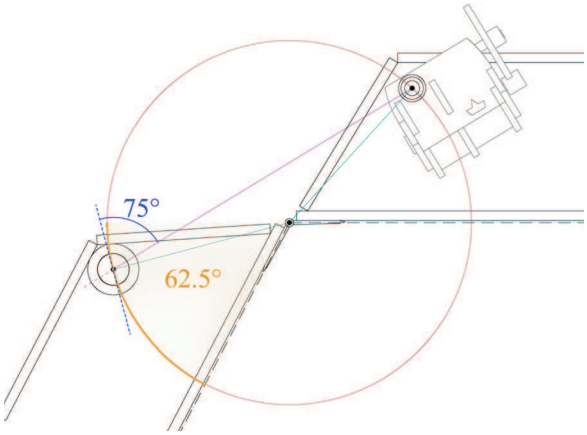

Fig. 4.27: Section of Kingfisher 1 in r^-

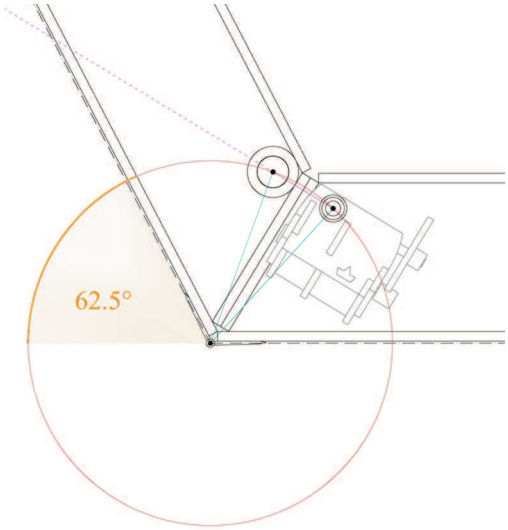

Fig. 4.28: Section of Kingfisher 1 in r^+

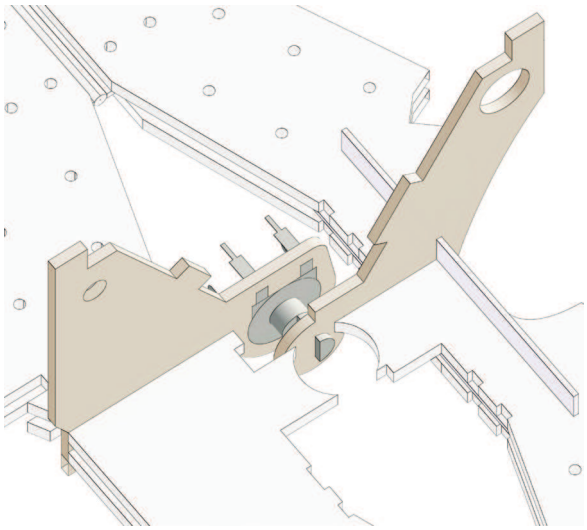

Fig. 4.29: Potentiometer mount

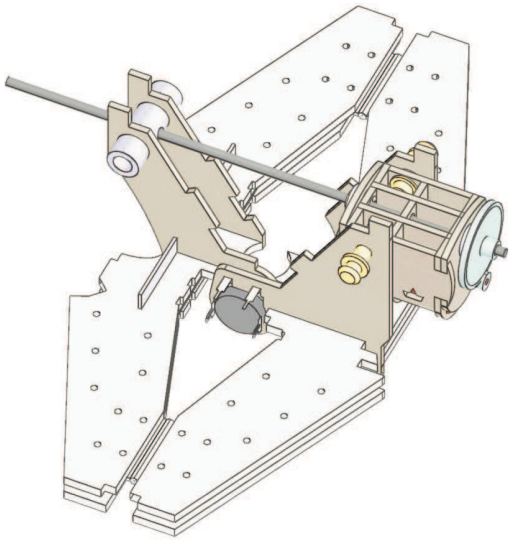

Fig. 4.30: Core MDF frame on base

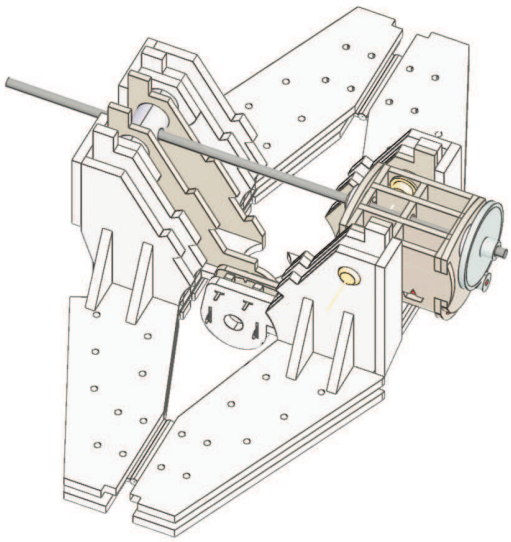

Fig. 4.32: Actuator without front shields

to ascertain the area taken up by the shaft. In large triangle configurations this wouldn't pose a problem, in small or narrow configurations, however, it does. Even after broad investigations the problem couldn't be solved, so that triangle configuration CAA (fig. 4.4) had to be removed from the set of options. Thus, in the triangulation of our start surface, the utilization of this configuration unfortunately has to be avoided. The design of the actuator's non-motorized side had to be adjusted to avoid intersections in any other configuration. This led to a very narrow part in the center of its baseplate and the use of an aluminum spine to reinforce the slender foam board.

Further testing proved the actuator design sufficient. It was ready to be produced in a large amount. Figures 4.33 to 4.35 show interconnected sets of Kingfisher 1. Video clip *"12_140128_KF1.avi"* in the appendix shows one of its first trials.

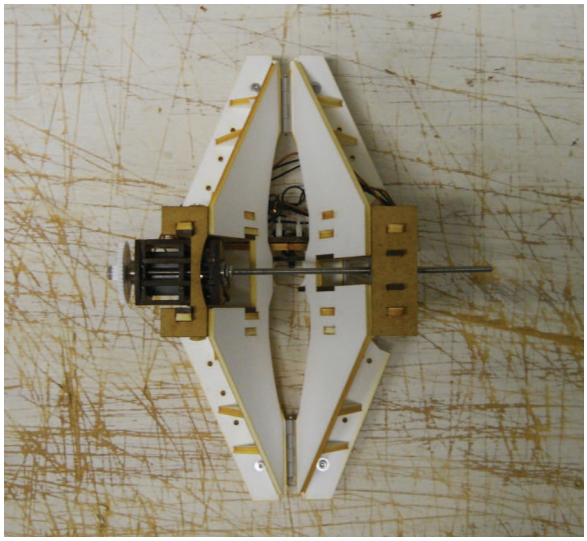

Fig. 4.31: Single agents

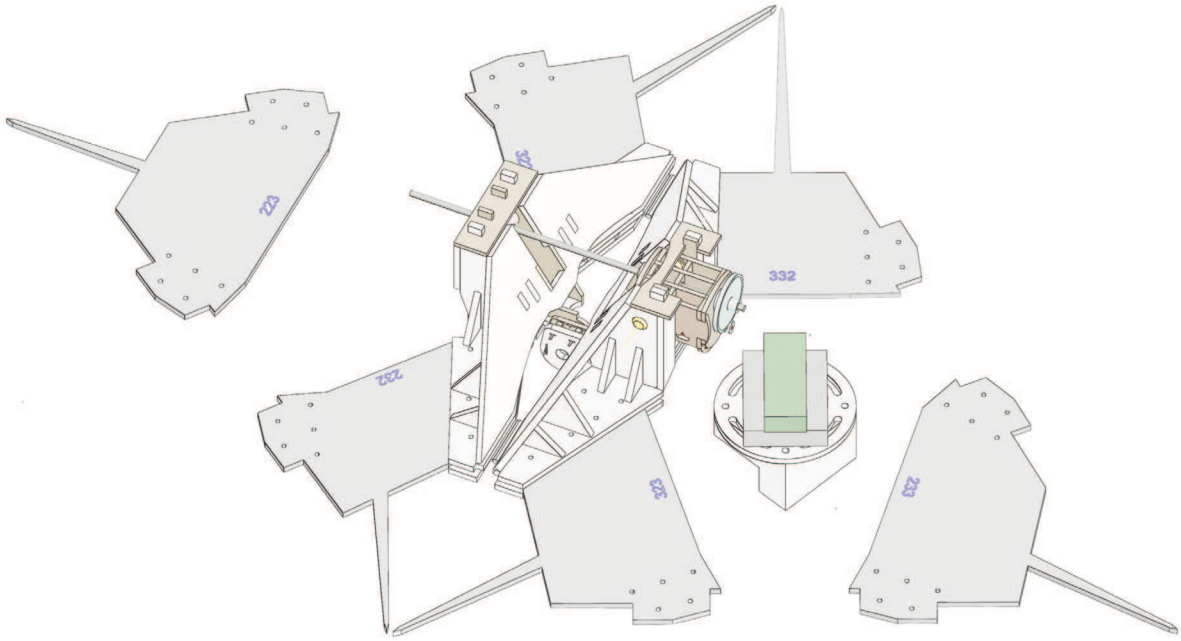

Fig. 4.33: single actuator with base connectors of configurations CBB neighboring BCC and control unit

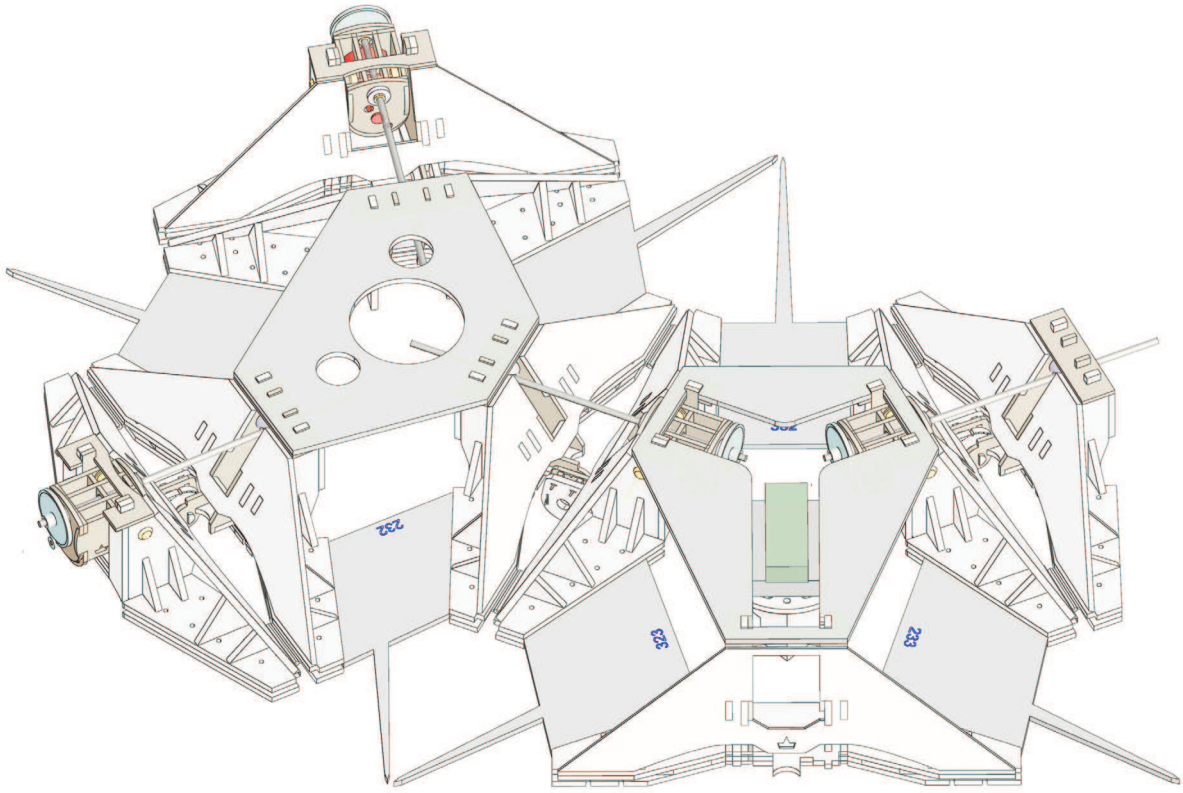

Fig. 4.34: complete setup of configurations CBB neighboring BCC

light gray:	base connectors and upper connectors	white:	PU foam cardboard sheet
green:	control board	brown:	MDF
blue:	code for correct assembly	red:	motor

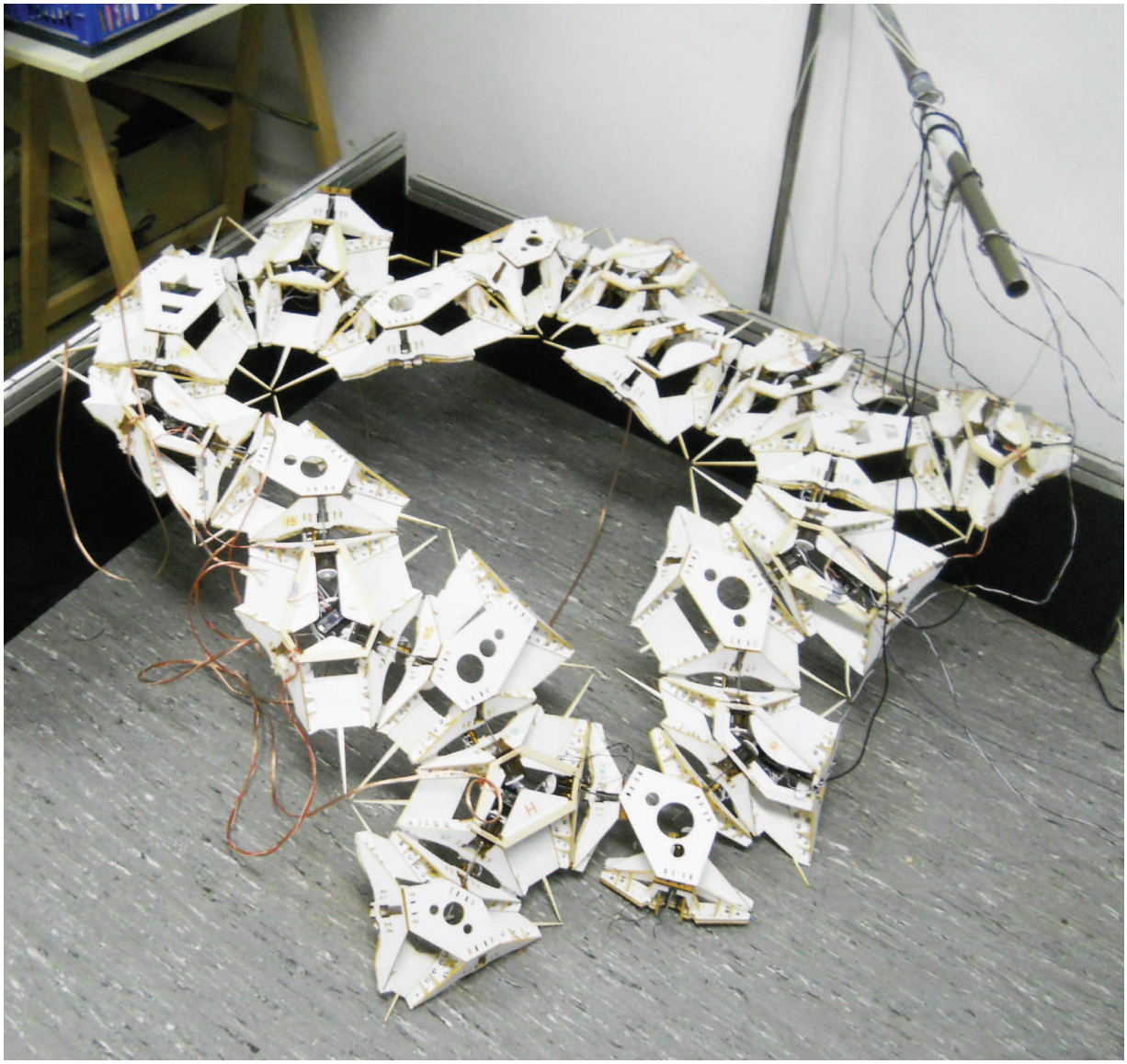

Fig. 4.35: Kingfisher 1 assembled

4.6 Electronic setup

In the following I will list the respective electronic parts used in the different versions of Kingfisher and, if necessary, add some comments on their performance. The choice of these pieces emerged from try and error, the main focus was to use the most powerful motor and to supply it with sufficient energy, but also to avoid the control boards to be harmed by current repulses caused by the motor's spool.

Kingfisher 0.1

motors:	Motraxx X-Slot Race 143, along with a Murata ceramic capacitor 1 µF
potentiometer:	Piher PT-15 10 kΩ
control:	Arduino Duemilenove
power supply:	control current through the Arduino (5V)

Kingfisher 0.2

motors:	Motraxx X-Slot 10S Race, Motraxx X-Slot 283 Tuning, Murata Z5U-5 1 µF
potentiometer:	Piher PT-15 10 kΩ
control:	Arduino Duemilenove, Arduino Micro
power supply:	12V 1A
motor driver:	L293E two channels, up to 600mA each (heated up and managed to melt the breadboard it was mount on)

Kingfisher 1

motors:	21 x Johnson 20543, Murata Z5U-5 ceramic capacitor 1 µF (21 times)
potentiometer:	21 x Piher PT-15 10 kΩ (21 times)
control:	1 x Arduino Duemilenove (to start and stop the simulation, connected to Firefly), 1 x Arduino Micro, 10x SparkFun Pro Micro 5V 16MHz In order to reduce cost, not every agent got one control board for itself. Instead, one control board controls two motors. In the code, however, their circuits run completely independent.
power supply:	2x LPK2-23 400W power supply, 12V 22A each
motor driver:	12 x L298N two channels, up to 2A each

4.7 Programming

Most of the tests on the way to the final prototype were run to prove the sufficient performance of the motors and their respective drivers in the matter of strength. In this sense it was advisable to simply use the Firefly plugin for Grasshopper in Rhinoceros 3D. Since this plugin supports Arduino models “Uno”, “Duemilenove” among others, but not the “Micro”, or SparkFun “Pro Micro” which were intended to be used in large amounts in the experiment, it was necessary to bridge this gap sending signals to the Duemilenove, the to the Micro and then to the motor in Kingfisher 0.1 and Kingfisher 0.2. In the final version, however, this plug-in was solely used to send an on/off signal to the micro controllers mount in the model, which execute the actual control code. This code was established step by step, by testing sub issues and combining them to a programming in the very last step.
The largest hurdle in this process was to apply a control curve for the actuators angles with simple DC motors. The potentiometers installed to constantly measure this angle, give a feedback on which the code reacts.

The control curves were obtained as follows:
In a previously executed simulation in Grasshopper, a unique index was assigned to every inner edge of the mesh. Using the Kangaroo engine which updates the positions of the vertices in every time step of the simulation, the angles over each edge were stored. The simulation consists of 1030 frames, thus we get 1030 angle values for each of the 22 inner edges of the mesh. These angles were measured in radians.
The potentiometer utilized, however is capable of rotating 270°. Its output is a 10 bit binary number, meaning it returns a number between 0 (resp. 0° rotation) and 1023 (resp. 270° rotation). Thus its resolution is about 0.26°. Note that, as described in 4.5 the third prototype provides a flexibility range form r^{-} = -62.5° to r^{+} = 62.5°. When we express the rotation of the agents as the10-bit output of their potentiometers we get 511 (1024/2-1) as a 0° rotation, r^{-} is equivalent to 274 and respectively r^{+} to 785.

The very first rule in the source code is to stop moving further, if the potentiometer returns a value that exceeds this range and only allow the agents to rotate in the other direction. This is to prevent the frame from cracking and to avoid unnecessary motor stress.

In the next step the control curve of each agent was translated into a list of 10-bit numbers. To avoid imprecisions in the potentiometers’ measurements due to inaccurate mountings of their shafts, their output at a rotation of exactly 0° was read and noted down. Then this value was used to adjust the control curves respectively. The source code of Kingfisher 1 is very simple. In simple terms it says:

1. Read the very first value of your control curve.
2. Chose from the following:
   - If the potentiometer’s output is below that value, move up
   - If the potentiometer’s output is above that value, move down
   - If the potentiometer’s output is equal to that value, move on to the next value in the control curve.
3. Return to 1.

4.8 Discussion

The actual goal was to build a physical prototype and to supply it with a code setup that enables it to fold itself into a desired, predefined shape. Furthermore, the goal was for the robot to do so without a central control, without the knowledge of an exact procedure, but with tools and rules to communicate and interactively reach this goal without cracking (see 3.3), avoiding jams and system inherent constraints. This creation of a very simple form of intelligence, however, couldn't be realized in the given amount of time. The actuators operate referring to a previously elaborated, implemented control curve. They simply repeat a predefined choreography. What could be realized, however, is the conception of a prototype that is feasible in the mechanical aspect and furthermore complies with the preset conditions of simplicity and standardization. It provides the sufficient strength to lift itself and the connected neighbors off the ground, a sensor to tell its actual position and the requirements to be able to communicate with its neighbors. The electronic setup reached a stage where different codes can be tested and give a direct, unbiased feedback of its feasibility. So to say the physical body is ready now, yet the non physical mind needs to be refined. The next step, however, must be to teach this system to teach itself. That is: Provide each agent with as less information as possible and promote it to interact with its neighbors, not only in the physical, but also in an informational manner.

5. Conclusion - Is this architecture?

The fashion in which buildings are erected nowadays surely represents one of the most complex procedures mankind has ever established. When we think of huge construction sites, especially in the developing countries, the coordination of thousands of workers, subcontracted supply and heavy equipment interlocking in the most synchronized and efficient way, it seems obsolete to do a research about robotic agents collaborating, to achieve a goal beyond their individual capabilities. This is exactly what building construction has been all about all along, but with the "agents" being workers instead of kooky threaded-shaft-perforated-mesh-robots. What's the use? The aim of this work was to find a correlation between a biological process, abstract its logic and translate it into tools of modern technology to suite an architectural purpose - the generation of shape. Furthermore it aimed to test the approach of design research. Physical model versus virtual simulation. Pure thought versus manual model making.
During my thesis work the by far most frequently asked question by my fellow students was: "So, do you still study architecture?", "To which exact architectural usage is your research?" To be honest, I don't have an exact answer to that. As mentioned, an imaginable application could be a flexible concrete formwork construction kit, but the main personal impellent behind this research was pure curiosity and the conviction, that collaborative organisms provide a tremendous potential to improve architectural day-to-day business. We just need to find a way to tap it. I certainly can not answer the question, whether my approach is feasible or *useful*, but I believe that we, as architects and engineers, can only gain innovation from going off beaten tracks and cooperate with representatives of other expertises. Thinking "outside the box". In that sense I would like to close with the words by Werner Sobek, one of the world's leading civil engineers, when asked about the collaboration between architects and civil engineers and the present situation in their respective research fields [12]:

"Aber ich glaube, eine Universität hat die noble Aufgabe, jenseits der Anwendungsforschung zu forschen."
("But I believe, that a university has the noble duty, to research beyond applied research.")

References

[1] A. Colorni, M. Dorigo, and V. Maniezzo. Distributed Optimization by Ant Colonies. In F. J. Varela and P. Bourgine, editors, Towards a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life, pages 134-142. MIT Press, Cambridge, MA, 1992.

[2] D. M. Gordon. 1999. Ants at Work: how an insect society is organized. Free Press, Simon and Schuster. 2000 paperback, W. W. Norton.

[3] Chacao, Caracas, Venezuela: Vertical Gymnasium Chacao, Pilot Project; Urban Think Tank 2004

[4] Aranya: an approach to settlement design. Planning and design of low-cost housing project at Indore, India. Ahmedabad: Housing and Urban Development Corporation/Vastu-Shilpa Foundation 1990

[5] “Play van Gendthallen! The making of the Freezing Favela”, Play the City, Amsterdam (NL), http://www.playthecity.nl/14859/en/the-making-of-the-freezing-favela (found: 2014/01/20)

[6] Julian Adenauer/Jörg Petruschat: Prototype! - physical, virtual, hybrid, smart - tackling new challenges in design and engineering, form+zweck, 2012, page 80-87

[7] Alexander I. Bobenko, Yuri B. Suris, “Discrete Differential Geometry: Integrable Structure”, Graduate Studies in Mathematics, Vol. 98, AMS, 2008, page 27-28

[8] Eric Bonabeau, Marco Dorigo, Guy Theraulaz, “Swarm Intelligence - From Natural to Artificial Systems”, A volume in the Santa Fe Institute Studies in the sciences of complexity, 1999, Oxford University Press, page 14-23

[9] Tachi, Tomohiro, Simulation of Rigid Origami, University of Tokio, 2009, http://www.tsg.ne.jp/TT/cg/SimulationOfRigidOrigami_tachi_4OSME.pdf (found: 2013/11/24)

[10] S. Belcastro and T. Hull. A mathematical model for non-flat origami. In Origami3: Proceedings of the 3rd International Meeting of Origami Mathematics, Science, and Education, pp. 39–51, 2002.

[11] Computer Graphics Group, The half edge data structure, Project OpenMesh, RWTH Aachen, http://openmesh.org/Documentation/OpenMesh-2.0-Documentation/mesh_hds.html (found: 2013/11/30)

[12] Julian Adenauer/Jörg Petruschat: Prototype! - physical, virtual, hybrid, smart - tackling new challenges in design and engineering, form+zweck, 2012, page 12-37

[13] Marvin Minsky, “The Society of Mind”. Simon and Schuster, New York. March 15, 1988

[14] Antonio Dimasio, Descartes’ Error: Emotion, Reason, and the Human Brain, Putnam, 1994; revised Penguin edition, 2005

[15] Bernd Pastuschka, Werner Sobek, Volkwin Marg, Interview über das Verhältnis von Architekt und Ingeneur, HafenCity Hamburg, 2012, http://www.youtube.com/watch?v=wFm8U-rhTIA (found: 2013/12/27) time: 15:18 - 16:19

Picture credits

[a] Arnhart, Larry, “The Darwinian Biology of Aristotle’s Political Animals,” American Journal of Political Science 38 (May 1994): page 464.

[b] Ciju, Cherian, Solent News, That’s teamwork: Ants turn themselves into a bridge after their route home is blocked, by Lyle Brennan, Daily Mail UK, 2012/03/24 , http://www.dailymail.co.uk/news/article-2119826/Indian-ants-turn-bridge-route-home-blocked.html#ixzz2r3swu6xF (found: 2014/01/15)

[c] Gent, Liza: Coreopsis Bud Unfolding. Last August Flowering Moments. Waterloo, Belgium 2007/08/11 http://forums.mooseyscountrygarden.com/pictures/last_photos_of_the_august_garden__coreopsis_bud_unfolding3_605.jpg (found: 14/01/07)

[d] Highsmith, Carol, Carol M. Highsmith’s America, Library of Congress, Prints and Photographs Division. US Airforce Acadamy Chapel, Colorado Springs, USA, Skidmore, Owings and Merrill,1962

[e] Origami in textile concrete, 5th Handler Convention on Light structures in Concrete, Jan Dirk van der Woerd, Rotislav Chudoba, Josef Hegger, Institute of Solid Construction, RTWH Aachen, 2012

[f] Servo motor “Lego NxT Robotics”, http://www.weston.org/schools/ms/technology/robotics/pages/servos.htm (found: 2014/01/07)

[g] Johnson Motors, “Johnson 20543” http://www.pollin.de/shop/dt/OTA1OTg2OTk-/Motoren/Gleichstrommotoren/Gleichstrommotor_Johnson_20543.html (found: 2014/01/07)