How to Design Morphologies. A Design Process for Autonomous Robots

Vincent Rist(✉) and Manfred Hild

Berliner Hochschule für Technik (BHT), Berlin, Germany
ristv@yahoo.com, manfred.hild@bht-berlin.de

Abstract. We describe a design process for robots in order to study the relationship between the agent's morphology and its behavioral capabilities. The limbs are modeled using circular arcs, resulting in smooth, round body shapes. From those two-dimensional models, a directed graph is generated, which represents the morphological manifold and captures all statically stable postures of the robot. On this graph, motion sequences can be found using abstract key postures properties that are morphologically independent. In a further step, we are investigating the extent to which our virtual 2D morphology is also suitable for a physical 3D robot. For this purpose, 2D morphologies are extruded linearly in the z-dimension so that they can be manufactured with a conventional 3D printer. We can then deploy the motion sequences on a physical robot, assembled using a convenient 3D-printable plug'n'clamp kit. The evaluation showed that with this process, one can either optimize a behavior for a specific morphology or try to optimize the morphology for a specific behavior. Even though the process does not yet find the most efficient motion sequence and more research will have to be conducted on that matter, a stable forward motion could be generated for every robot morphology.

Keywords: Internal models and representations · Dynamical systems approaches · Autonomous robotics

1 Introduction

The morphology of an agent mostly stays constant throughout its lifetime, even though it heavily influences the agent's behavioral diversity. This is due to the time and money consuming venture of designing, assembling and deploying a fully functioning robot and due to the difficulty of engineering a morphologically flexible machine. Those are also the reasons why the study of morphology has usually been constrained to simulation, which allows for enormous amounts of morphologies to be genetically optimized. Even though the artificial evolution has proven to produce diverse morphology and suitable behavior [1,10], the approach has two downsides: Firstly, because of its constraint to simulation it is only partially applicable to the physical world [2]. And secondly, it is difficult to reason why a found solution is optimal, which means changing the task at hand

L. Cañamero et al. (Eds.): SAB 2022, LNAI 13499, pp. 3–14, 2022.
https://doi.org/10.1007/978-3-031-16770-6_1

usually requires a new evolution. The proposed design process tries to address these downsides.

The morphology and its physical properties turn embodied robots into complex dynamical systems. The postures corresponding to the stable fixed points of these systems create manifolds inside the high-dimensional sensorimotor state space [6,7]. Any state a robot might find itself in is represented as a point on the manifold, and any path along the manifold that connect two of those states corresponds to a motion sequence. Hence, the manifold of a particular robot holds information about all possible motion sequences that it can pursue. They can be explored dynamically and autonomously during runtime [3] and offer an alternative method of planning action than to simulate the body shape and its physical properties onboard in real-time.

2 Proposal of a Design Process

By using the morphological manifold for motion planning, the consequences of changing a morphological parameter can be directly visualized in real-time. The intent of the proposed design process is not necessarily to produce faster, cheaper or more efficient robots, but to have a tool which helps robot designers to reason about their design decision based on the shape of the manifolds and the resulting motion sequences. We created a modular plug'n'clamp kit which allows the designers to reduce the time and resource expenses of assembling and deploying robots, which facilitates rapid evaluation of the morphology and behavior in the physical world. The four-part process is schematically visualized in Fig. 1. The following subsections discuss each part in more detail.

2.1 Arc Representation

Commonly, morphologies are modeled using polygons in 2D or polyhedrons in 3D, which produce shapes with flat edges and sharp corners. We decided to extend our mathematical model to allow for smooth and round contours as well. This means robots could have both flat limbs, like the hooves of ungulate animals, or round limbs, like the paws of felines. Adding smooth, round contours to the pool of possible morphological features has some advantages: The first being

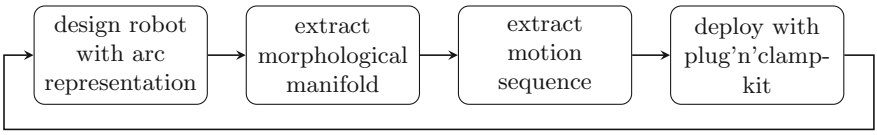


Fig. 1. Schematic visualization of the proposed design process. Morphologies have to be geometrically designed, are analyzed for their morphological manifold and motion sequences and are finally physically deployed with a plug'n'clamp kit. Experiments in the real world deliver insights on how to upgrade the robot design and the next iteration of the process begins.

that flat edges and sharp corners are features of ideal shapes that are never perfectly implemented due to manufacturing constraints, e.g., the nozzle diameter of a 3D-printer. Secondly, smooth shapes are commonly found in nature. Robots with a similar aesthetic might seem more organic and less mechanic to the observer. Thirdly, robots that fall onto a flat edge absorb the kinetic energy of the impact, which might damaged it. A round edge, however, can redirect this energy by rolling off. Robots might even use their round edges to induce rocking by periodically shifting their center of mass, i.e., robots with round limbs can experience interesting dynamical behavior.

One way of mathematically describing smooth, round contours is with the help of circular arcs. A circular arc A is a continuous subset of a circle outline C that is defined by its radius r and its midpoint $m = (m_\mathrm{x}, m_\mathrm{y})$:

$$A \subset C = \{(x, y) \in \mathbb{R}^2 \mid r^2 = (x - m_\mathrm{x})^2 + (y - m_\mathrm{y})^2\}$$

By tangentially concatenating those arcs one can model parametrized shapes like seen in Fig. 2. Those arc shapes are particularly well suited as two-dimensional limb models for several reasons: First off, the arc representation requires fewer parameters to model a smooth and round limb than the approximation of the same shape with a polygon or with splines. Additionally, the radius of the arc touching the ground can be used to adjust the friction, an important morphological property for many behaviors like grasping or locomotion. And finally, the continuous property of the arc representation also causes the morphological manifolds to be continuous, whereas the discrete polygon representation creates discontinuous steps in the manifold corresponding to postures where the robot tips over the corners of the polygon.

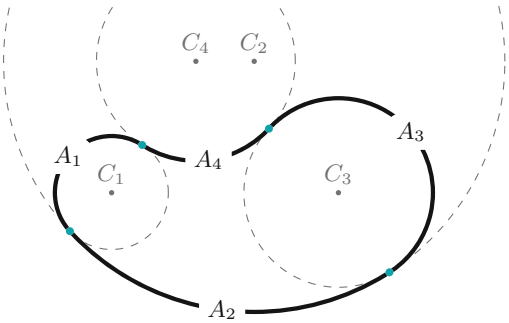


Fig. 2. Example of a limb modeled with four circular arcs A_1 through A_4. The transition from one arc to the next has to be tangential to avoid corners. This particular shape can be defined unambiguously by four parameters: the radii of C_1 and C_3, the distance between their midpoints, and the central angle of the connecting arcs, which is the same for both A_2 and A_4. See [9] for more detail.

We use the arc representation to model two-dimensional robot limbs that can have both the advantages of sharp, flat and of smooth, round contours.

2.2 Morphological Manifold as a Directed Graph

The posture of a robot is defined by its current body configuration, i.e., its degrees of freedom, and by the orientation of the robot in the world. Let us inspect *Tablebot*, the robot from Fig. 3(a). It consists of three limbs: the two legs are connected to the torso by rotary joints. The joints are indicated by black circles in the figure. Tablebot's center of mass is visualized as a black dot, whereas the white dots symbolize the centers of mass of each individual limb. The joint angles φ_l and φ_r define the relative rotation between the legs and the torso, i.e., the configuration. A third angle φ_o specifies the rotation of the torso relative to the ground, which defines the orientation of the robot in its two-dimensional world. For (a) $\varphi_\mathrm{o} = 0$ holds, which is why it is omitted in the figure.

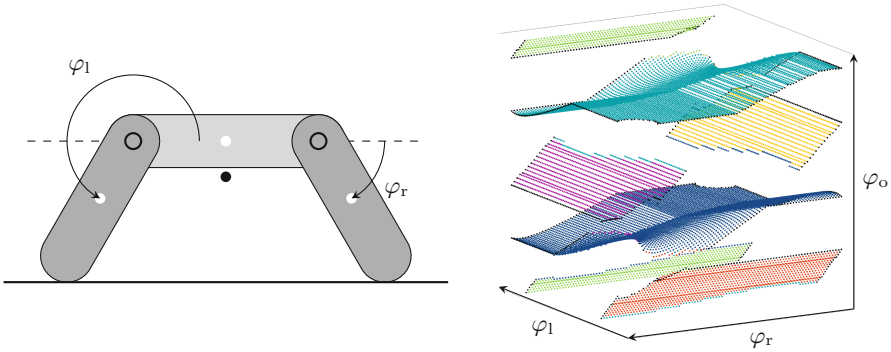


Fig. 3. (a) Side-view of *Tablebot*, a two-dimensional robot with three limbs. (b) Morphological manifold of Tablebot. Each colored dot indicates a node of the directed graph. The resolution is 80 nodes per axis. (Color figure online)

The robot's orientation is subject to a dynamical system driven by gravity. In essence, it behaves like this: If the orthogonal projection of the center of mass to the ground lays within the convex hull of the points or areas touching the ground, a stable fixed point attractor is reached. Otherwise, the robot falls, which means it experiences a change in orientation. The stable fixed point attractors of this system correspond to stable postures of the robot. In the case of (a) the x-coordinate of the center of mass lays between the points of ground contact, so the posture is stable. With its degrees of freedom, the robot can manipulate its own ground contact and its center of mass, and thus also the orientation of the stable postures. Considering all possible stable postures, they form a manifold inside the posture space, just like the one in Fig. 3(b).

Even though some parts of the manifold can be formulated as geometrical planes it becomes difficult to describe the manifold analytically. [4] shows how robots can autonomously learn quadrics that represent their manifold during runtime. An alternative approach is to use a discrete model of the manifold structured as a directed graph. The nodes of the graph correspond to discrete stable postures. If a robot that currently holds posture a can transition into posture b by moving its joint by a certain angle, the postures are connected by a directed edge: $(a \rightarrow b)$. The edge has to be a directed one because not all posture transitions are reversible, e.g., if the joint movement caused the robot to fall. This is the reason why the morphological manifold commonly consists of a number of disconnected submanifolds, which are distinguished by color in Fig. 3(b). All postures that are part of the same submanifold are bidirectionally connected, i.e., no falling occurs during transition, if the movement speed is not too fast. The irreversible fall-edges and physical constraints like angle bounds or self collision make up the borders of the submanifolds.

2.3 Algorithmically Determined Motion Sequences

Manifold graphs hold information about all possible stable postures and how (and if at all) the robot can transition between them. Each posture has certain properties, like the positions of the center of mass, of the ground contact points, etc. This data structure enables algorithms to find suitable key postures on the graph and connect them to create motion sequences for certain tasks. In this study, we chose a locomotion sequence consisting of only three key postures. For example, one might want to generate a simple locomotion sequence by periodically transitioning between the three key postures in Fig. 4: The robot starts in posture (a) where the distance between its ground contact points is relatively small. By positively actuating its right leg, a transition to posture (b) is accomplished. Since most of the weight (black dot) is now shifted to the left side, the right leg is more likely to slide across the ground than the left leg. In the transition from (b) to (c) the robot shifts its weight onto the other side while at the

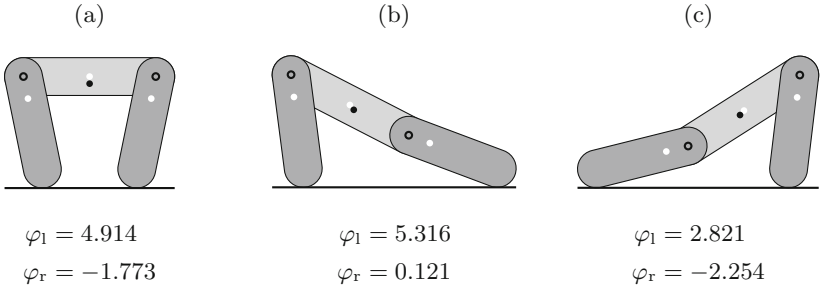


Fig. 4. Three key postures for locomotion. By periodically transitioning from (a) through (c) the robot will slide from left to right.

same time keeping a fixed distance between the utmost contact points with the ground, which avoids slipping of the feet. Lastly, the robot intentionally reduces this contact distance by actuating its left leg. The shift in weight will now cause the left leg to slip, and the robot finds itself in posture (a) once again. This coordination of friction and slippage will cause the robot to move forward. The motion resembles a human gait when moving across a slippery surface.

The two posture properties at play here are the distance d between the points of ground contact and the relative weight position w. If the robot splits its legs outwards as far as possible, then $d \approx 1$ and for $d \approx 0$ the points of contact move closer together. $w = 0$ if the center of mass is right on top of the left point of contact and $w = 1$ for the right side, meaning $w = 1/2$ corresponds to an even weight distribution. Figure 5 shows a clipped projection of the cyan colored submanifold from Fig. 3(b) onto the configuration space. The color gradient of the displayed images shows the value of d and w of the corresponding posture. Key posture (a) needs to have a low d and a balanced $w = 1/2$, which is why postures closer to the bottom right corner are well suited. (b) and (c) on the other hand, need postures with a high d value and strongly shifted w (i.e., $w \approx 0$ or $w \approx 1$), which can be found in the upper right or bottom left corner respectively.

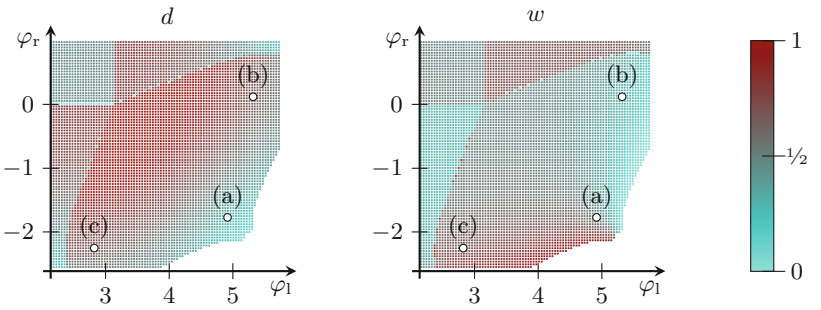


Fig. 5. A projection of the manifold onto the configuration space with a resolution of 90 postures per axis. The color gradient on the left indicates the value of d (distance between points of ground contact), whereas the right one indicates w (relative weight position). Dark red colors correspond to higher and bright cyan colors to lower values. (Color figure online)

The postures in Fig. 4 were found using a gradient search. Using a function f to determine how much similarity a posture shares to the key posture, the graph can be traversed by iteratively moving to a better neighboring posture x_{n+1}:

$$x_{n+1} = x_n + f'(d(x_n), w(x_n))$$

Starting from a posture x_0 this method will find a local maximum. The functions and starting postures x_0 were hand-tuned. This approach leaves room for optimization, but it is a step towards abstracting behavior from morphology through posture properties.

2.4 Plug'n'Clamp Kit

The final step of the design process is assembly and deployment. Through modularity robots can become morphologically flexible machines even during runtime [5]. Inspired by the toy-robot-assembly-kit *Topobo* [8] we created a similar modular plug'n'clamp kit. The limbs that are modeled with the arc representation are extruded linearly in the z-dimension, and then directly converted into printable STL-files, which helps robot designers to reduce time and effort during assembly and deployment of their agents. The kit complies with the following criteria:

1. *Printability*: Limbs and connecting components are designed for optimal 3D-printing. This means that filament and time expenses are reduced, no supporting structures are needed, and predetermined breaking points are avoided.
2. *Pluggability*: The effort involved with assembly and disassembly remains low. Robots can be assembled without screws, tools, or ball bearings by using a plug and clamp system instead.
3. *Modularity*: All modules can be combined with all other modules. The kit tries to constrain the morphology as little as possible.
4. *Simplicity*: The number of different individual parts needed to enable deployment is reduced to a minimum. Standard parts such as the pins can be reused for other morphologies and do not require reprinting.
5. *Aesthetics*: The kit prefers rounded contours over sharp corners and straight edges so that the assembled robots nicely harmonize with the arc representation.

The essential components of the kit are displayed in Fig. 6. The little H-shaped pins (a) are used as universal connecting tools, which can be printed within five minutes. Robot limbs such as (b) consist of two symmetrical parts, which are printed separately and are then plugged together. The advantages of using this method is that firstly only the frame and some connecting structures have to be printed, which reduces filament consumption and printing time. Secondly, motors or other peripherals can be clamped in between the two halves as done in (b) avoiding the need for screws. Here we were using the *Dynamixel XL330-M288-T* motors from *Rotobis*. Plugging multiple limbs together will look like (c), a 3D embodiment of the 2D Tablebot morphology.

3 Evaluation

Two different test series have been conducted to evaluate the utility of the described process. The *OpenCM9.04* from *Robotis* served as a microprocessor and four AA-batteries in series created a supply voltage of 6 V. All other parts were printed with gray PLA-filament, resulting in 279 g of total robot weight. With both legs stretched outward and the torso limb touching the ground, the robot reaches a length of 39 cm. The robot was placed on a medium-density fiberboard plate and then executed three cycles of a given motion sequence,

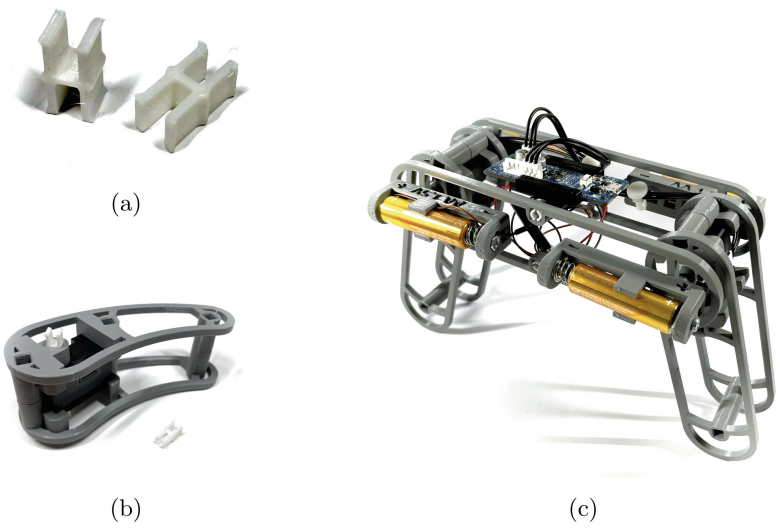


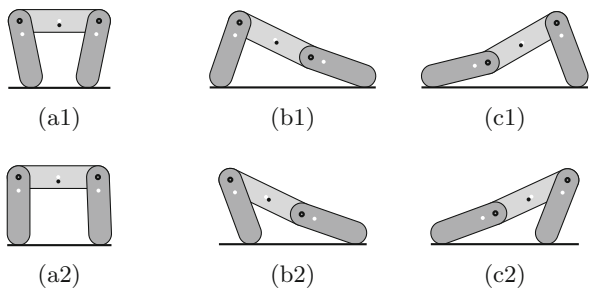
Fig. 6. (a) Universal, H-shaped pins that are used to manually connect and disconnect two components. (b) A limb consisting of two separately printed parts with a motor clamped in-between. (c) A fully assembled and deployable embodiment of the Tablebot morphology with a processor and batteries.

which itself consisted of three key postures. Then the average walking distance per cycle D and the average energy consumption per cycle E was measured.

For the first test series, we described four different motion sequences for Tablebot by tweaking the key posture properties as shown in Fig. 7. Sequences that include posture (a1) caused the robot to pull its legs closer together as compared to (a2). When applying (b1) and (c1) the robot made a big step by maximizing d and by putting less focus on w. For (b2) and (c2) it was the other way around: less d and more w. Each sequence was then evaluated seven times on the robot.

The average results are presented in Fig. 8. They demonstrate that for this morphology, taking big steps seems to increase D and strongly shifting the center of mass seems to decrease E. Motion sequence S2 gets the most walk distance out of its energy consumption. This test series demonstrates how the design process provides a method of optimizing a motion sequence for a certain task and a certain morphology.

As the morphology stayed constant for the first test series, we wanted to see if the key posture properties also apply to different morphologies. So we randomly scattered the shape parameters of Tablebot's legs inside the parameter space and received the four new morphologies seen in Fig. 9. The same torso module was used for the four robots as it had little influence on the task at hand and to avoid reprinting additional modules. Each morphology M_i was analyzed for a motion sequence S_i by using the same key posture properties as sequence S2



S1	(a1)	(b1) (c1)
S2	(a1)	(b2) (c2)
S3	(a2)	(b1) (c1)
S4	(a2)	(b2) (c2)

Fig. 7. Six different key postures. The table on the right combines the postures into motion sequences S1 through S4.

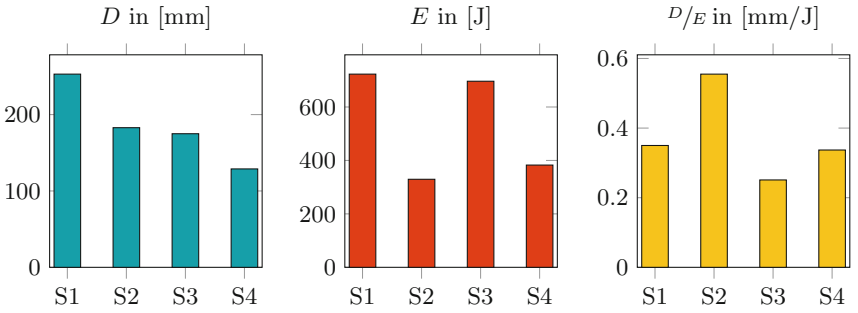


Fig. 8. The averaged results from testing motion sequence S1 through S4 on the Tablebot morphology. D represents the walking distance that the robot travelled per cycle. E is the energy consumed for that motion.

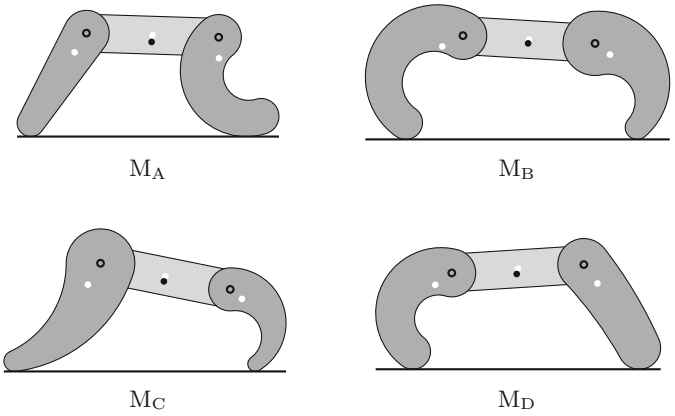


Fig. 9. Four new morphologies. They are constructed similarly to Tablebot, but with diversified leg parameters. To be comparable to each other, the overall robot length stays constant. The walking direction is still from left to right.

of the first test series. Each of the four sequences was then executed on every morphology, resulting in the confusion matrices of Fig. 10.

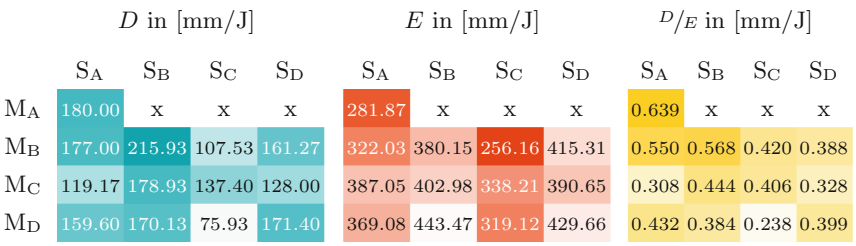


Fig. 10. Confusion matrices of the averaged measurements per cycle for the second test series. The rows represent the four morphologies, and the columns are the motion sequences that were found on their respective manifolds. The diagonals hence hold the matching sequence-morphologies-pairs. Strongly colored cells performed better than lighter ones. Cells containing an "x" mean that this sequence was unstable on the morphology and no meaningful values could be measured. (Color figure online)

The results show that the crab-like morphology M_B realized the fastest locomotion, but M_A had the most energy efficient motion sequence. Sequence S_D could move its matching morphology the furthest, but as it consumed a lot of energy, motion sequence S_A was the more efficient movement for M_D. This demonstrates how different walking behaviors can be designed with the correct key posture properties. Agents might casually prefer a slow but efficient way of moving forward, but in case of an emergency a faster and more energy consuming "sprint" might temporarily be activated. For M_C the process failed as sequence S_C was neither the fastest nor the most efficient option for the robot. This could be due to the fact that in contrary to the other robots the length difference between the two legs was significantly larger, which caused its manifold to be very dissimilar to the rest. The key posture properties might have not applied in the same way as for the other robots. More research will have to be conducted. For M_A, none of the other sequences worked: the flat hind leg (left leg) would get turned too far underneath the torso, causing the robot to easily fall on its "butt". A round-edged hind leg like on M_B or M_D prevents such constraints, once more emphasizing the benefits of the arc representation. Video recordings of some of the reported test series, like displayed in Fig. 11, can be streamed from our cloud[1], and for a more detailed walkthrough of the whole process consult [9].

[1] https://cloud.bht-berlin.de/index.php/s/RpHTdf4mLLEByJW.

Fig. 11. Selected screenshots of the video recordings. Here Morphology M_B is executing motion sequence S_B, which was the fastest morphology-motion-pair from the second test series.

4 Discussion and Outlook

In this study, we only looked at 2D quasi-static motion on flat terrain to keep the dimensionality of the manifolds low. This limits the range of possibly generated gaits to the sliding motion we could observe in this paper. Increasing the dimensionality of the agent increases its complexity, and hence it diversifies the range of possible gaits or motion sequences. Transferring the robot into a three-dimensional world, adding more limbs, or enabling dynamic motion are all potential areas of future research, but introduce new dimensions to the system and make it harder to model morphologies, abstract behavior, and generate 3D-printable parts for the plug'n'clamp kit. By adding the derivatives of the static dimensions (φ'_o, φ''_o, and so on) to the systems, the momentum of the limbs could be taken into account, allowing for the generation of more dynamical gaits, like galloping. It would be interesting to study how non-flat environments manifest themselves inside the manifold, and if the agent could use these changes to classify the obstacles in order to adapt the previously discovered motions accordingly. But increasing the dimensionality of the manifolds also impairs their visualizeability, which make it harder for a designer to reason his design decisions. The curse of dimensionality strikes again.

Optimization algorithms could be used to automatize the process of finding gaits or other motion sequences in the morphological manifold superseding, the current hand-tuned gradient search.

Also, we plan to have the agents explore their behavioral possibilities autonomously in the physical world by dynamically generating their manifold graph as done in [3] and by learning their motion sequences themselves.

Our overall goal, is to build an interactive tool that can visualize the relationship between morphology and behavior, ultimately helping the designer to comprehend the consequences of his decisions. Once the correlations are better known, the tool might even and make reasonable design suggestion. But there is still work and research to be done.

References

1. Bongard, J.: Morphological change in machines accelerates the evolution of robust behavior. Proc. Natl. Acad. Sci. **108**(4), 1234–1239 (2011). https://doi.org/10.1073/pnas.1015390108

2. Brodbeck, L., Hauser, S., Iida, F.: Morphological evolution of physical robots through model-free phenotype development. PLoS ONE **10**(6), 1–17 (2015). https://doi.org/10.1371/journal.pone.0128444
3. Hild, M., Kubisch, M.: Self-exploration of autonomous robots using attractor-based behavior control and ABC-learning. In: Eleventh Scandinavian Conference on Artificial Intelligence, pp. 153–162. IOS Press (2011)
4. Hild, M., Kubisch, M., Höfer, S.: Using quadric-representing neurons (QRENs) for real-time learning of an implicit body model. In: Proceedings of the 11th Conference on Mobile Robot and Competitions (2011)
5. Hild, M., Siedel, T., Benckendorff, C., Thiele, C., Spranger, M.: Myon, A New Humanoid, pp. 25–44. Springer, Boston (2012). https://doi.org/10.1007/978-1-4614-3064-3_2
6. Höfer, S., Hild, M., Kubisch, M.: Using slow feature analysis to extract behavioural manifolds related to humanoid robot postures. In: Tenth International Conference on Epigenetic Robotics, pp. 43–50 (2010)
7. Peters, R., Jenkins, O.C.: Uncovering manifold structures in Robonaut's sensory-data state space. In: 5th IEEE-RAS International Conference on Humanoid Robots, pp. 369–374. IEEE (2005)
8. Raffle, H.S., Parkes, A.J., Ishii, H.: Topobo: A constructive assembly system with kinetic memory. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 647–654. CHI 2004, Association for Computing Machinery, New York, NY, USA (2004). https://doi.org/10.1145/985692.985774
9. Rist, V.: Algorithmische Morphologien für autonome Roboter. Bachelors thesis, Berliner Hochschule für Technik (2022)
10. Sims, K.: Evolving 3D morphology and behavior by competition. Artif. Life (1994). https://doi.org/10.1162/artl.1994.1.4.353