Horizon 2020

Societies of Symbiotic Robot-Plant Bio-Hybrids as Social Architectural Artifacts

Deliverable D2.2

Report on the final algorithms and plant-affection of bio-hybrid organism

Date of preparation: 2017/03/31	**Revision: 1 (r732)**
Start date of project: 2015/04/01	**Duration: 48 months**
Project coordinator: UPB	**Classification: public**
Partners:	
lead: **AMU**	*contribution:* **UPB, UNIGRAZ, CITA**
Project website:	http://florarobotica.eu/

H2020-FETPROACT-2014

DELIVERABLE SUMMARY SHEET

Grant agreement number:	640959
Project acronym:	*flora robotica*
Title:	Societies of Symbiotic Robot-Plant Bio-Hybrids as Social Architectural Artifacts

Deliverable N^o:	Deliverable D2.2
Due date:	M24
Delivery date:	2017/03/31

Name:	**Report on the final algorithms and plant-affection of bio-hybrid organism**
Description:	Development of algorithms and experimental approaches for bio-hybrid systems are reported. We introduce the Vascular Morphogenesis Controller (VMC) to direct the growth of artificial structures. Another controller allows to steer a bean's tip towards user-defined spatial targets. The presented control methods are combined with modeling techniques that apply to plant growth and artificial growth and hence provide a good basis for a general design methodology for bio-hybrid growth systems. We report experiments with light, chemicals, and vibrations as examples of stimuli to influence plants in desired ways (motion, shape, function). Light of various wavelengths is used in the Plant Binary Decision Wall (PBDW) experiment, where autonomous robotic nodes use LEDs and proximity sensing to influence and interact with plants. The reported results are an important stepping stone of the project as they provide the basic methodology to develop bio-hybrids systems of natural plants and robots.

Partners owning:	**AMU**
Partners contributed:	**UPB, UNIGRAZ, CITA**
Made available to:	public

Contents

1 Introduction

Our main objective in *flora robotica* is to create plant-robot bio-hybrids. In the following we report results that are key for effective bio-hybrids. We have made progress in developing controllers for plant shaping, we have worked on procedures for plant stimulations and tip detection for the Plant Binary Decision Wall (PBDW), we report models for self-organized bio-hybrid growth, and advances of investigations of plant affection by different stimulations.

In order to establish more robust bio-hybrids, that could also survive in different environments and could consist of different plant species, we require various ways of influencing plant growth and development. In Section 2 we present an experimental setup to investigate the influence of different chemical and mechanical stimulations on plants.

Plant-robot communication is crucial in bio-hybrids. We have noticed unexploited potential of interorgan rapid communication in plants and report an investigation of long-distance signaling pathways (see Section 3).

Shaping plants is an important tasks for the robotic components of the bio-hybrid. In Section 4 we report how we use light stimuli in the PBDW setup and how we implement the proximity sensing of plants. We focus on a task that allows a user to specify a target for the plant to approach and the robotic system then grows the plant towards that target.

In Section 5 the Vascular Morphogenesis Controller (VMC) is reported in all details. This is our main approach to direct the self-assembly and growth of artificial components in the bio-hybrid. In Section 6 we conclude and make a few comments about future tasks.

While here we focus on software and wetware, more information about our hardware approach can be found in deliverable *D1.2 Evaluation of mechatronics prototype of the robotic symbiont including supporting software*.

Figure 1: Photo of *A. thaliana* that were grown in drought conditions, that is, without watering for one week. 10 mM SA of water (control) was applied in spray once a day.

2 Investigation on plant stimulation

2.1 Application of chemicals and influence on plants growth and development

During the vegetative season of 2016, two types of phytohormones were applied: auxin (1mM NAA) and cytokinin (4.5 uM kinetin); both to tomatoes and hornbeams. Effects of spraying phytohormones on one of the tomatoe branches were shortly described already in PTR Part B. The same phytohormones in spray were applied once a week on single branches of hornbeams, which were growing in a greenhouse. Effects due to weak light conditions, that strongly promotes the elongation of all branches, were not taken into consideration. Moreover, a persistent aphids infection has occurred with a great impact on the trees. In autumn 2016, the hornbeams were transferred to the AMU Botanical Garden, where we have executed an efficient treatment. The subject of Biogenic Volatile Organic Compounds (BVOC) was studied for influencing plant growth and survival in *flora robotica*. Chemically BVOC are terpenoids, derivatives of amino acids or fatty acids that are emitted by plants in stress conditions. One of the BVOC is salicylic acid, that is also a phytohormone responsible for resistance against pathogens and for inducing necrotic reactions after infection. 1 mM and 10 mM salicylic acid (SA) was applied as spray for 5-weeks to *Arabidopsis thaliana* in an early generative stage of development (see Fig. 1). Watering was withdrawn and then every day the spraying was applied. No increase in the plant survival rate during this period of water shortage was observed after applying SA, as expected for high SA levels [47]. However, high SA levels result also in oxidative bursts and necrotic effects. Stress due to the SA application might in these cases outplay the protective effect. Lower SA concentrations are already tested for *Arabidopsis*. For the vegetative season of 2017, we have developed our plan for the application of phytohormones and BVOC. Most of our experiments have already started on March 1, 2017: bean, pea, and cucumber have been sown. Poplars

Figure 2: Poplars *Populus tremula x alba* after transfer to greenhouse from growth chamber (March 1, 2107). In order to induce branching, tips of the trees were cut at least one time before transfer to greenhouse.

were transferred from the growth chamber to the greenhouse (see Fig. 2). The chosen plant species are: poplar (*Populus tremula x alba*), common bean (*Phaseolus vulgaris*), *Arabidopsis thaliana*, pea (*Pisum sativum*), and cucumber (*Cucumis sativus*). Poplars are fast growing trees in which the DR5:GUS transgene was introduced. DR5:GUS enable staining tissues with higher auxin levels, so additional information could be derived by using this line. Although poplars are known for their low quality of wood (so for *flora robotica*'s construction tasks they might not be the best species), poplars probably will tolerate more greenhouse conditions than beech and hornbeam. Bean, pea, and cucumber are creeper vegetables which grow fast, positively react to scaffolds (braids) and it is possible to go through a few of their life cycles in one year. The range of phytohormones that have been prepared for use are: auxin (IBA, the more stable form than NAA), cytokinin (6-BAP, kinetin), and gibberellic acid A3. BVOCs that are in use for influencing plants: isoprene, ocimene, and farnesene. Phytohormones and BVOC are applied in spray and by sponges (see Fig. 3). The sponge actuation is performed by fixing the sponge, that is saturated with chemicals, on the plant's stem. The effect of different chemicals and different ways of applications will be compared with each other and with a negative control (water). We are focusing on plant shapes and growth. Although the general picture of phytohormone effects on plants growth is well-known, previous research did not focus on shaping plants by localized applications of chemicals. Besides plant shapes, the physiological parameters will be tested using physiological sensors. The monitored parameters are: chlorophyll content of leaves, transpiration rate measured by stomatal conductance and thermography. The appropriate devices (Decagon SC1, DualexScientific+, Flir E6) are going to be set up in April 2017.

2.2 Mechanical stimulation on plants

The biomechanical behavior of plants in response to vibrations has not been described in all details yet [9]. Moreover, vibrations have got an influence on growth, shape, and wood structure. In 2016 we have shown the non-destructive effect of vibration motors on the growth of beans. For 2017, the effect of vibration motors on the growth of woody plants (poplar) is under preparation.

Figure 3: Application of BVOCs (1% farnesene, 1% hexanol, 1% ocimene) using sponges (currently a semi-automated application is implemented).

Five vibration motors were placed on a branch of poplar. The motors are controlled by the MU Board and a Raspberry Pi. The individual branches of poplars are subject to various vibration modes. Vibrations will be individually chosen in every case. They will vibrate the branch in a way that minimizes the spread of vibration to the stem or other branches.

An air flow produced by a fan has being used to influence a tree's shape and wood structure. The influence of this "artificial wind" is more global than in the case of using vibration motors. However, a strong air flow also results in vibrations of the plant organs. Some of the poplars were subject to "artificial wind" training in a growth chamber (see Fig. 4). Now in greenhouse conditions, these plants are subject to applications of chemicals. One of the trees did not survive mechanical stress (training) and a broken tree top is an example of the potential destructive power of mechanical stimulation (see Fig. 5).

Braided structures, which have become an essential part of *flora robotica*, have to be considered also as mechanical stimulation for plants. The growth of beans on braids was shown previously, now the growth of perennial plants on a braid is investigated. We study the change of mechanical properties of braid-plant systems as a result of interweaving living plants with braids. Stiffness (measured by a force perpendicular to the surface that is required for a certain displacement) is a feature of interest. For this purpose we reconfigure a Zwick/Roell Material Testing System. Early tests of mechanical properties of complete braid structures were not satisfactory (see Fig. 6), hence, additional experiments will be done. Plant species that are going to be tested are: *Fallopia*, common bean, *Clematis*, and *Ipomoea*.

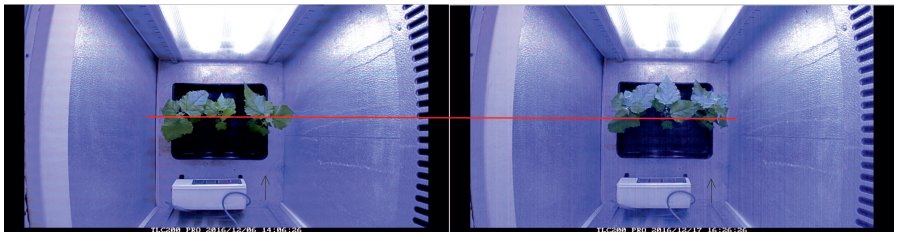


Figure 4: (left) poplar at the beginning of the wind training; (right) after two weeks of the application of constant air flow. The light source was placed opposite to the air flow source, in order to strengthen the effect of directional growth.

Figure 5: Comparison of a poplar that was broken by air flow and one of the survivors. The big difference in the root system sizes is clearly visible; the roots significantly determined the shoots' resistance to wind.

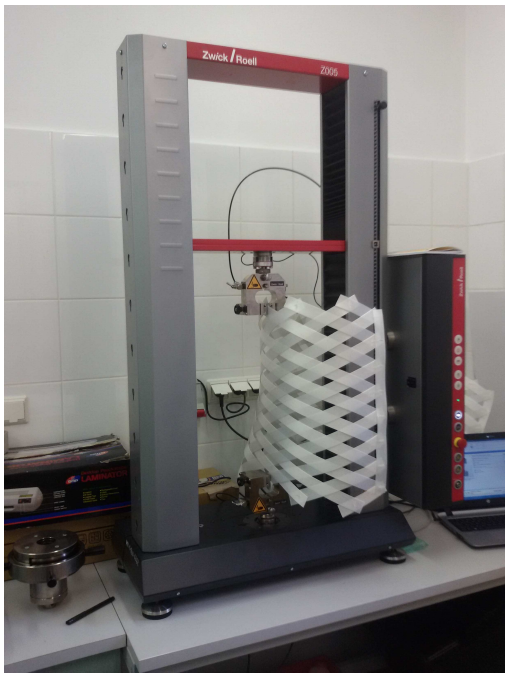


Figure 6: One of the preliminary tests on mechanical properties of braided structures with a Zwick/Roell Material Testing System.

3 Plants intercellular signaling pathways for bio-hybrid communication

3.1 Introduction

Intercellular signaling differs between plants and animals. In animals there are cells specialized in conducting electric signals, neurons, and centralized nervous system, that ensure a fast and efficient transmission of signals and the integration of multiple signals within the brain. These structures form the basis for the development of advanced bionic prostheses, an example of existing bio-hybrids. Neurons could be directly stimulated by electrodes. The brain could learn the meaning of these received signals, train itself in the perception of the signals and the creation of appropriate reactions [26]. Plants are sessile and lack both neurons and nervous systems, so different approaches for creating bio-hybrids have to be developed. In plants many mechanisms of rapid long distance signaling were shown, such as four types of electric signaling, calcium waves, and Reactive Oxygen Species (ROS) [21]. All of them are involved in responses to biotic and abiotic stresses. In order to establish plant bio-hybrids in the future, a deeper investigation of plant signaling pathways is required. Deciphering signaling crosstalk of plants in parallel with development of new sensors will result in novel possibilities for creating plant bio-hybrids [37]. Our current progress of studying plant long distance signaling is presented in the following.

3.2 Crossings of Arabidopsis thaliana lines

The *Arabidopsis thaliana* line with overexpression of Jas9-VENUS [25] is used as sensor line that reveals the effect of rapid long distance signaling by imaging changes in the level of jasmonic acid (JA). A decrease in VENUS fluorescence means that JA levels rise (Fig. 7). H2B-RFP give a fluorescence reference that should show a stable intensity level. This line was used to prove changes in the JA level after mechanical damage in a distant organ [25]. Effects of local stimulation, that are important for a whole organism, could then be observed in distal tissues or organs. Such processes are also important for plant bio-hybrids and for understanding of how the reaction of plants can help in establishing plant-robot communication. Jas9-VENUS is now being crossed with different *A. thaliana* knockout lines: *rbohd, tpc1-2, glr3.3, glr3.6, mca1, mca2* (Fig. 8). These knockout lines are NASC lines deprived of key genes involved in ROS – RbohD, calcium – MCA1, MCA2, electric signaling GLR3.3, GLR3.6, so some mechanisms of long distance signaling should be deprived or even absent. Investigation of long distance signaling alterations would be feasible after obtaining crosses of knock-out lines with the sensor line. Synthesis of jasmonic acid occurs often after contact with stress factors, in nature it is correlated with events triggered by herbivores or pathogens. By the end of February, crossings of NASC knockout lines – N645413, N577608, N535353, N570610, N239204, N669029 – with the Jas9:VENUS line were done. Seedlings of first generations are growing and will be genotyped soon. For the genotyping, primers have already been designed. Polymerase for direct PCR, REDExtract-N-Amp PCR Kit (Sigma-Aldrich) will be used.

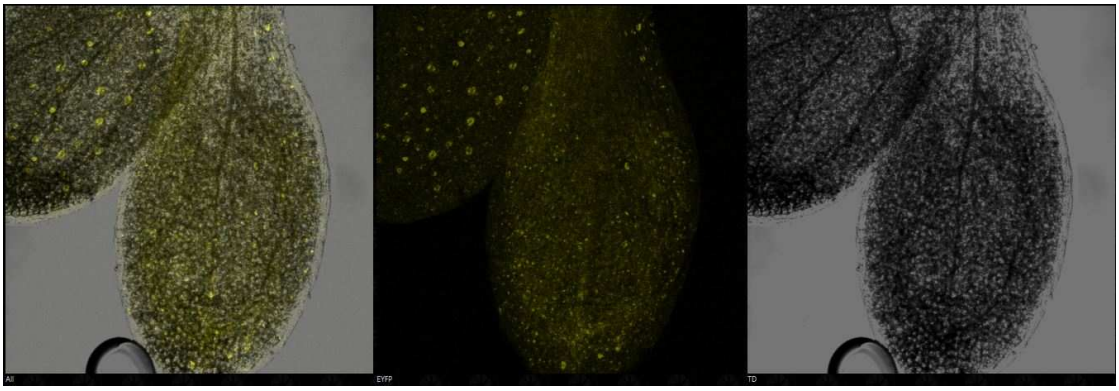


Figure 7: Image of Jas9-VENUS in the *A. thaliana* seedling. The nuclear localization of Jas9 is visible. Maximum Intensity Projection from stack image is shown. Left: fluorescence, excitation 514 nm, emission 520-560 nm; right: transmission image; middle: merged image.

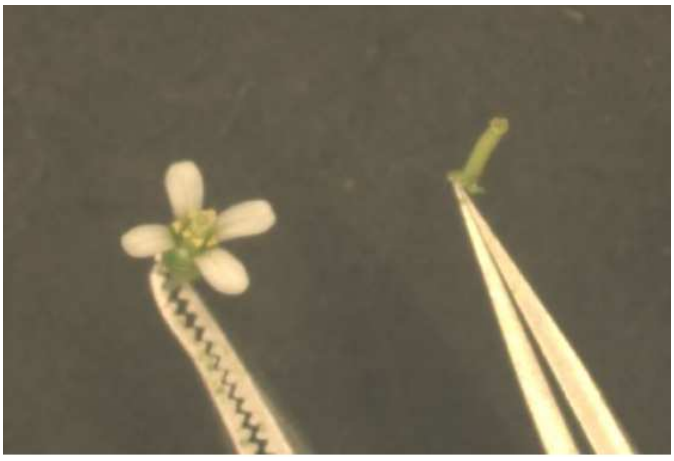


Figure 8: Image of *A. thaliana* flowers: natural (left) and after gymnoecium isolation (right). Gymnoecium isolation is required to preventing self-pollination. After gymnoecium isolation, pollen from the paternal anther is transferred to gymnoecium.

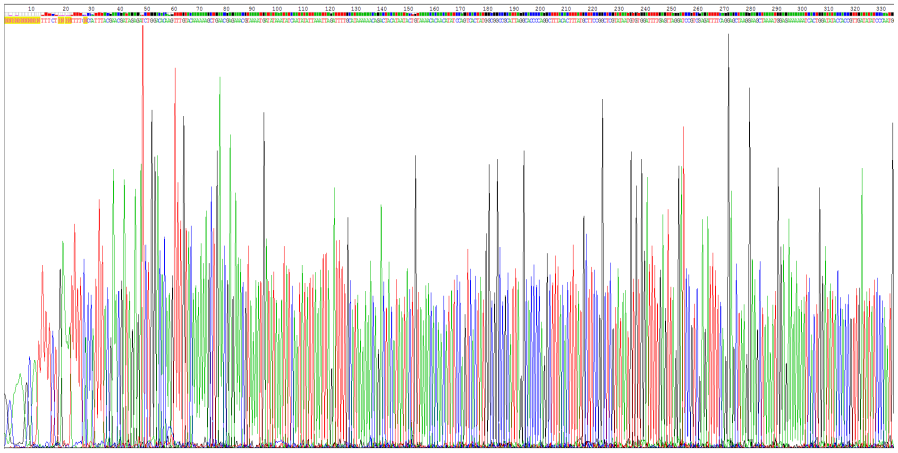


Figure 9: Part of sequencing results of the RbohD:pENTR dTOPO DNA construct. Next, the sequence was analyzed with the Nucleotide BLAST online tool to compare it with database sequences.

3.3 Cloning of genes of interest

Cloning of genes involved in different long distance signaling is done for investigating protein-protein interactions. First, RNA was isolated from 4-weeks old *Arabidopsis thaliana* rosette leaves and cDNA was synthesized (Super Script II Transcriptase, Thermo Scientific). Then, primers specific for genes of interest (i.e., RbohD, MCA1, MCA2, GLR3.3, GLR3.6) were used for their amplification with Phusion Polymerase (Thermo Scientific). Amplified genes were cloned into pENTR dTOPO (Invitrogen) with the help of NcoI, AscI restriction enzymes (Fast Digest, Thermo Scientific) and ligase T4 (Promega). The cloning products were sequenced (Fig. 9) and in the next step cloning into expression plasmids from pSAT and pSITE series was done. Chosen expression plasmids make it possible to transiently overexpress a gene (i.e., protein as a gene's product) in plant cells, such as protoplasts or epidermis. The gene of interest is expressed together with a fluorescence protein tag on N- or C- terminus, so protein localization would be obtained. Moreover, protein-protein interactions will be studied with the FRET-FLIM microscopy method and with co-immunoprecipitation. Information about interactions between proteins involved in different signaling pathways would help us in deciphering crosstalk between different long distance signaling pathways. In the beginning of March 2017, genes of interest were cloned into expression plasmids. The preparations for overexpression in *Arabidopsis thaliana* protoplasts and tobacco epidermis are in progress and at least preliminary results will be obtained within 2017.

4 Robots for shaping natural plants

Here we describe the methods for two distinct efforts for the robotic shaping of natural plants. One of these efforts is evolved control of plant growth and motion (refer to our paper [18]), and the other is distributed robotic control through the Plant Binary Decision Wall (PBDW) setup (refer to our paper [45]).

4.1 Light stimulus

4.1.1. Phototropism in a simple bio-hybrid setup

As described in [18], in comparison to many other living systems, plants are slow in many of their activities including, of course, their growth. For example, the common bean plant (*Phaseolus vulgaris*), which is considered to be a fast growing plant, grows on average 3cm per day [4]. In addition to growth, plants also show motion, which is often ignored due to its low speed. Bean shoots' intrinsic motion (circumnutation [40]) allows the plant tips to explore their local environment and – together with phototropism (i.e., directed growth towards or away from light [6]) – influence growth to approach more preferable regions. Plant motion seems underestimated by many, likely because their speed is very slow in relation to time scales of human perception. However, on these slow time scales of plant activities, the speed of motion is still considerably faster than the speed of growth. For example, according to our preliminary experiments, bean plants (longer than 20cm) bend towards a light source with a velocity of up to 4.4mm/min. Angular velocities of the intrinsic circumnutation reported in literature are even larger [29].

In our bio-hybrid approach, the robot needs to detect the plant's position to allow for closed-loop control. The robot also needs to impose appropriate stimuli at appropriate times to influence the plant in the desired way. We investigate how the motion and growth of a plant can be influenced by a robotic hardware setup that uses light as an attractive stimulus. The light sources in the setup are controlled by a closed-loop controller detecting the plant's tip and reacting to its position such that the tip reaches a number of target points. For that, we first create an appropriate simulator that addresses relevant features of the plant growth based on the data collected from a set of preliminary experiments with the plant and the light sources controlled by a predefined open-loop controller.

4.1.2. Phototropism and 'shade-avoidance' in the PBDW

As described in [45], here we use the climbing common bean (*Phaseolus vulgaris*) as the biological element. This plant exhibits strong positive phototropism (i.e., directional growth towards blue light) and climb the mechanical connections in our scaffold setup, greatly enhancing the probability of the plant's small growth tip being detected by an IR-proximity sensor. Flowering plants' active directional growth towards a light source (i.e., phototropism) is triggered by the stimulus of blue and UV-A bands (wavelengths $\lambda \in [340\text{nm}, 500\text{nm}]$) [6]. Plants' photosynthesis (i.e., building up energy-rich biomolecules using light energy) uses any frequencies of the visible spectrum $\lambda \in [400\text{nm}, 700\text{nm}]$, and the process is especially efficient under the red band $\lambda \in [620\text{nm}, 680\text{nm}]$ [19]. There are many additional health effects of light frequencies, as flowering plants, for example, require small amounts of blue light for healthy growth [7].

Even though green light is the most reflected visible band, any present can significantly impact the plant [41] in photosynthesis and potentially in the phototropism process. In short, green light is not required for either of these processes to occur, but it could affect either or both. The plenty watery and air-filled compartments scatter the light and while red and blue light are

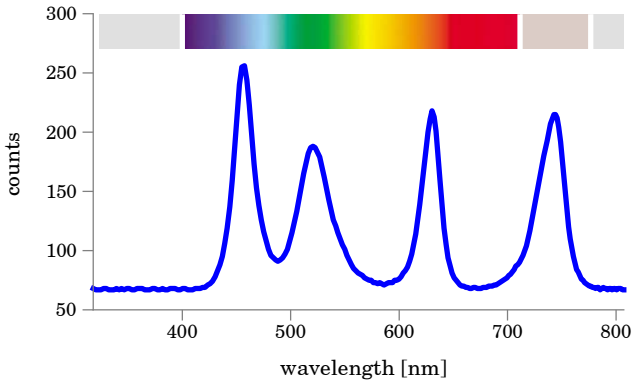


Figure 10: The spectrum of light emitted by the robots, peaks are blue, green, red (RGB LEDs), and far-red ($\lambda \approx 740$nm). Measured at distance of 20 cm, facing the robot (LEDs at intensity as in experiments).

excellently absorbed by the surface layers of the leaf, green light mainly drives photosynthesis in the leaves' interior. Here we exclude green light, to maximize the difference in wavelength of the cues (as stated above) and also to be able to run the investigated monochromatic LEDs at the highest possible intensity.

Natural plant reaction to the far-red band $\lambda \in [700\text{nm}, 800\text{nm}]$ stimulus is complex and varies greatly among species. In many species, combinations of red and far-red wavelengths can trigger the 'shade avoidance syndrome' [33]. The protein that, when activated by red light, mediates the developmental changes associated with this frequency band (Phytochrome) is in turn deactivated by far-red light. Plants efficiently absorb the red band of light, in contrast to far-red, which is mostly reflected by or passed through the plant. Thus, very low ratios of red-to-far-red *in vivo* can signal to the plant that it is shaded by closely neighboring plants. This stimulus typically leads to enhanced stem elongation and reduced shoot branching, as well as reduced investments in leaves and roots. The direction of incident blue light (indicating gaps in the canopy) and the direction of relatively high far-red (indicating competing plants as sources of future shade) are vital information for the plant's natural survival [42].

In our current experiment setup, the primary light used is in the visible spectrum $\lambda \in [400\text{nm}, 700\text{nm}]$, delivered by the robotic node's six RGB LEDs with light emission peaks at $\lambda \in \{625\text{nm}, 525\text{nm}, 465\text{nm}\}$ and a combined maximal dissipation of 3W. The range of emitted light is relevant to plants, as discussed above. The secondary light used is the far-red band $\lambda \in [740,745]$nm, emitting from three 3W far-red LEDs. In Fig. 10, the spectrum of emitted light by a robotic node is shown (as measured by a Hamamatsu C12666MA Micro-Spectrometer). The spectrometer was not calibrated with light sources of defined spectra, therefore, the vertical axis (labeled 'counts') gives arbitrary values that show relative intensities.

4.2 Proximity detection of plants

4.2.1. Plant detection via image sampling

Preliminary plant experiments. As described in [18], our *tip-motion model* builds on our prior work [44] by using the same group of photographed preliminary experiments to create a data set of plant growth and motion. These preliminary experiments (described here according to [44]) use a predetermined open-loop controller which simplistically alternates the two light

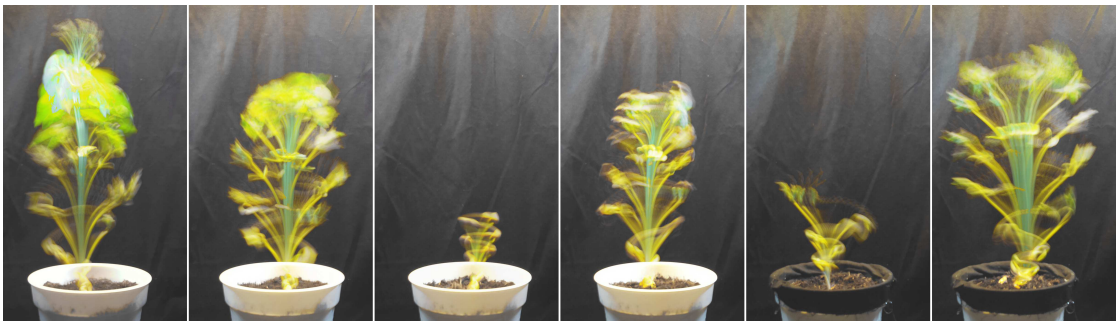


Figure 11: Compiled time-lapse photographs of experiments. From left to right, experiments 1, 2, 3, 4, 5, and 6. Photographs originally published in prior work [44].

sources over regular time intervals. There are six repetitions of the experiment. In each repetition, the light source alternates every six hours and the experiment is photographed at five minute intervals. The plants' tip position and stem geometry over time (see compiled images for each experiment in Fig. 11) show that their dynamics are substantially influenced by both growth and motion[1]. Despite the consistent setup conditions in each repetition, the compiled images show variety in the plants' patterns of horizontal motion and overall height. In plant science, variance in experiments is an expected phenomenon and is dealt with by conducting large quantities of repetitions. In our work, we instead take an engineering approach, and deal with plant variance by testing whether our method succeeds in controlling a plant despite certain unpredictability in behavior.

Tip detection. The images described above provide time-stamped raw data documenting plant responses to the predetermined open-loop controllers. The images are sampled[2] at 1/8 resolution and further processed by using the following method for detecting the plant tips. The tip-detection method works based on two sets of photographs. The first set contains the background photographs showing the setup without a plant. The second set contains the photographs from the preliminary experiments.

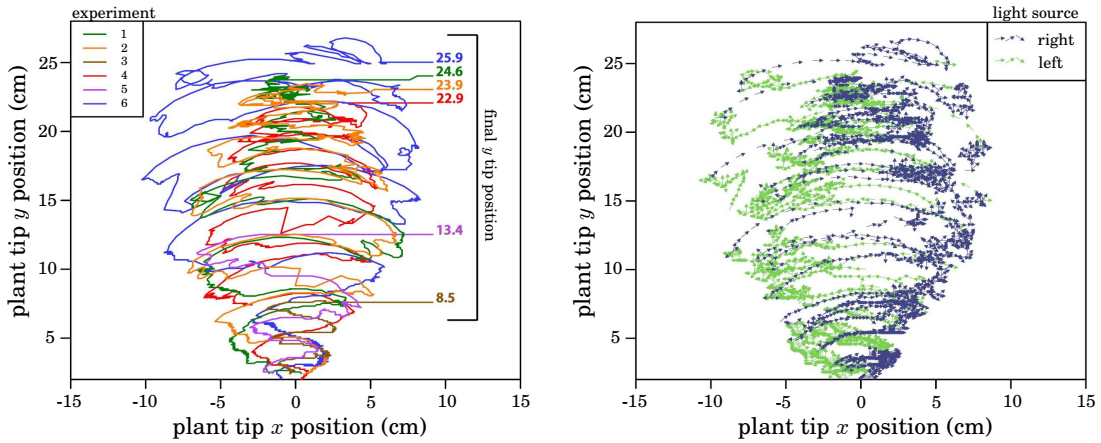
For the background photographs, many examples of each controller state are included in the set, as there can be slight variations in lighting conditions cast on the background. The green RGB channel value at each pixel position (i, j) is isolated and remapped onto the domain $(0.0,\ 1.0)$, resulting in a matrix $S_k = \{s_{i,j,k}\}$ for all images k. To represent the range of green values possible at every pixel in the background, matrices $A = \{a_{i,j}\}$ and $B = \{b_{i,j}\}$ are constructed with the same shape as S_n matrices. Every entry of A and B are the minimum and maximum values of the corresponding entries in all the S matrices:

$$\forall a_{i,j}, b_{i,j} : \quad a_{i,j} = \min_k(s_{i,j,k}), \quad b_{i,j} = \max_k(s_{i,j,k}). \tag{1}$$

After constructing these matrices representing the background setup, the photographs from the preliminary experiments containing plants are processed. The green channel value is again isolated for each pixel (i, j) and remapped to the domain $(0.0,\ 1.0)$, then is saved into the

[1] Find a video online: https://youtu.be/r4PknIwgTyo

[2] The tip-detection was implemented in two separate programming platforms. First, we implemented the method in Iron-Python and native libraries of the VPL Grasshopper pertaining to image sampling for processing the data in simulation. Then, we implemented the method in standard Python, utilizing the OpenCV library. This was then used for the reality gap experiments detailed in Sec. 4.3.4.



(a) Trajectories of plant tip position in the six preliminary experiments, with experiment number indicated by color (refer to Fig. 11) and height in the final time step in each experiment indicated on the right-hand side.

(b) Vector direction of Δx and Δy of plant tip trajectories at each time step, in all six preliminary experiments, with color indicating right or left light source.

Figure 12: Plant tip trajectories from preliminary experiments (only includes real data; does not include mirrored). Figures originally published in prior work [44].

matrices $G_k = \{g_{i,j,k}\}$ for every image k of the preliminary experiment. The G_k matrices are then compared against the range matrices A and B from the background setup in order to detect pixels containing plant material. A pixel (i, j) inside a certain cropped window is identified as containing plant material if its value is external to the corresponding range, with respect to threshold $\theta = 0.2$,

$$P_k = \{(i,j) \mid \quad \forall i,j: \quad g_{i,j,k} < (a_{i,j} - \theta) \vee g_{i,j,k} > (b_{i,j} + \theta)\}. \tag{2}$$

Each identified plant pixel is extracted to set P, and their $\mathbf{x}_p = (x_p, y_p)$ coordinate positions are used to identify two possible locations of the plant's tip. In order to locate the tip $\mathbf{x}_t$, plant pixels are compared to the globally defined anchor $\mathbf{x}_a = (x_a, y_a)$, representing the position where the plant stem emerges from the soil. We identify two possible tip positions (corner point $\mathbf{x}_c = (x_c, y_c)$ and high point $\mathbf{x}_h = (x_h, y_h)$),

$$\mathbf{x}_c = \underset{\mathbf{x}_p \in P}{\arg\max} |x_a - x_p| + |y_a - y_p|, \tag{3}$$

$$\mathbf{x}_h = \underset{\mathbf{x}_p \in P}{\arg\max} |y_a - y_p|. \tag{4}$$

From the two points $\mathbf{x}_c$ and $\mathbf{x}_h$, the one closer in Euclidean distance to the previously detected tip $\mathbf{x}_{t-1}$ is selected as the current tip $\mathbf{x}_t$,

$$\mathbf{x}_t = \begin{cases} \mathbf{x}_h, & \text{if } \|\mathbf{x}_{t-1} - \mathbf{x}_h\| < \|\mathbf{x}_{t-1} - \mathbf{x}_c\| \\ \mathbf{x}_c, & \text{else} \end{cases}. \tag{5}$$

The plant tip positions $\mathbf{x}_t$ are not based on physical measurement units but on pixels. We make use of this position data in our purpose-specific model described below.

The trajectories of tip positions are shown in Fig. 12(a), with the six experiments indicated by color. Fig. 12(b) shows the 2D vector direction of tip-motion at each time step in the experiments,

categorized according to active light source (right light as blue vectors and left light as green vectors).

10-point stem detection. Future extensions of the controllers will use 10-point descriptions of the full plant stem. For these extensions, we use the anchor point and tip point described above. The remaining intermediary points $((x_2, y_2), \ldots, (x_9, y_9))$ describing the stem are preliminarily identified from set P, and then smoothed using a method that preserves a faithful representation of stiffening dynamics. These eight intermediary stem points are defined as

$$y_i = \frac{i}{9}|y_1 - y_{10}| + y_1, \tag{6}$$

$$x_i = \overline{x_p} : \forall x_p \in P : |y_p - y_i| < \theta_2, \tag{7}$$

where $\theta_2 = 30$ pixels. In this way, we define a 10-point description of the full stem $(a, (x_2, y_2), \ldots, (x_9, y_9), t)$. Due to minor variations in images caused by shadows and light reflections on the stem, this 10-point detection contains some errors. We address these errors using a simple algorithm based on Smoothing via Iterative Averaging (SIA) [28], which preserves the key topological features of the curve being smoothed. Our algorithm utilizes the equation

$$(x_i, y_i) = \left(\frac{1}{2}(x_{i-1} + x_{i+1}), \frac{1}{2}(y_{i-1} + y_{i+1})\right) \tag{8}$$

iteratively, according to the following steps:

1. for $i = 2, 4, 6,$ and 8, apply equation 8,
2. for $i = 3, 5, 7,$ and 9, apply equation 8,
3. for $i = 2, 4, 6,$ and 8, apply equation 8.

In this way, the intermediate stem points are smoothed with the SIA-based process, while the tip and anchor point remain unchanged. We then convert the smoothed 10-point sets to cm and scale them to match the physical setup dimensions. The resulting data set (see Fig. 16) is used to construct our correlated data-driven model, described in the following section.

4.2.2. Proximity sensing in the PBDW

As described in [45], detecting the proximity of an approaching plant tip is challenging. In order to select the most successful option, we tested three proximity sensors. First, in an early phase experiment, we tested MPR121 capacitive touch sensors (see Fig. 14(b)). In this test, threads are mounted on a wall in a diagrid layout, for the plants to climb and grow along. Soldering tin is wound around the threads (close to diagrid intersections) and is connected to the capacitive touch sensors. In this test, the plants were correctly detected only about 50% of the time. Second, in an early prototype of our robotic node (see Fig. 14(b)), we tested both HC-SR04 digital Ultrasonic sensors and GP2Y0A41SK0F analog IR-proximity sensors. The ultrasonic sensors work most successfully when facing a flat object perpendicularly, maximizing reflections. Due to the changing orientation and small size of a plant growth tip, the sensor readings were very inaccurate. By contrast, the IR-proximity sensors achieved high accuracy. This is especially true when the direction of the approaching plant is predetermined (e.g., climbing a thread/rod), as is the case in our experiment setup (see Fig. 13). We therefore selected the analog IR-proximity sensor for the task of plant proximity detection. Based on several tests and given that a reading is taken every 5 seconds, we develop a weighted arithmetic mean approach to detecting a plant

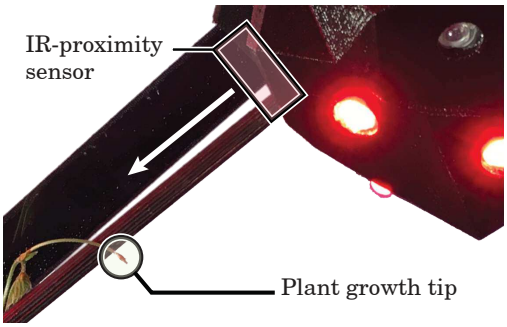
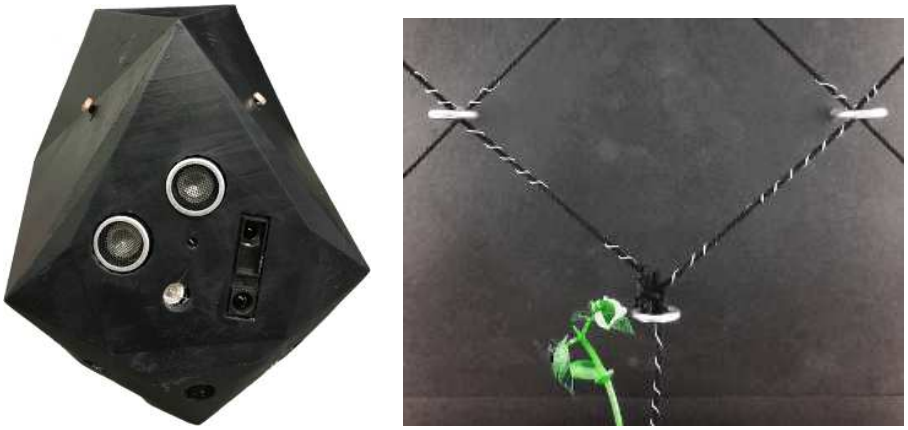


Figure 13: Plant tip detection.



(a) Early robotic node prototype.

(b) Capacitive touch experiment.

Figure 14: Earlier prototype and proximity sensor experiments.

with an IR-proximity sensor. Where the last sensor reading gives 20% of the final average weight, the plant tip is reliably detected at a threshold of approximately 5 cm distance from the node. According to the sensor datasheet, the emitted IR wavelength should be restricted to 870 +/- 70 nm, however, if we measure the light spectral distribution with a spectrometer from the immediate vicinity of the IR source (at a distance of $<$ 2mm) we can detect some emission of critical wavelengths below 800 nm, slightly overlapping the action spectrum of the far-red LED. However, no critical wavelengths overlapping the LED spectrum are measured at distances $>$ 2mm. Therefore, although the plant response to far-red light heavily depends on the ratio to incident red light, the spectrum emitted from the sensor can be safely assumed to not interfere with the far-red LEDs acting as stimulus.

4.3 Control algorithms

4.3.1. Introduction to evolved control for user-defined targets

As described in [18], our evolutionary robotics approach to the task of steering plant growth and motion follows the methods described in this section. First, the bio-hybrid system setup shown in Fig. 15 is used in both preliminary data-gathering experiments and reality gap experiments.

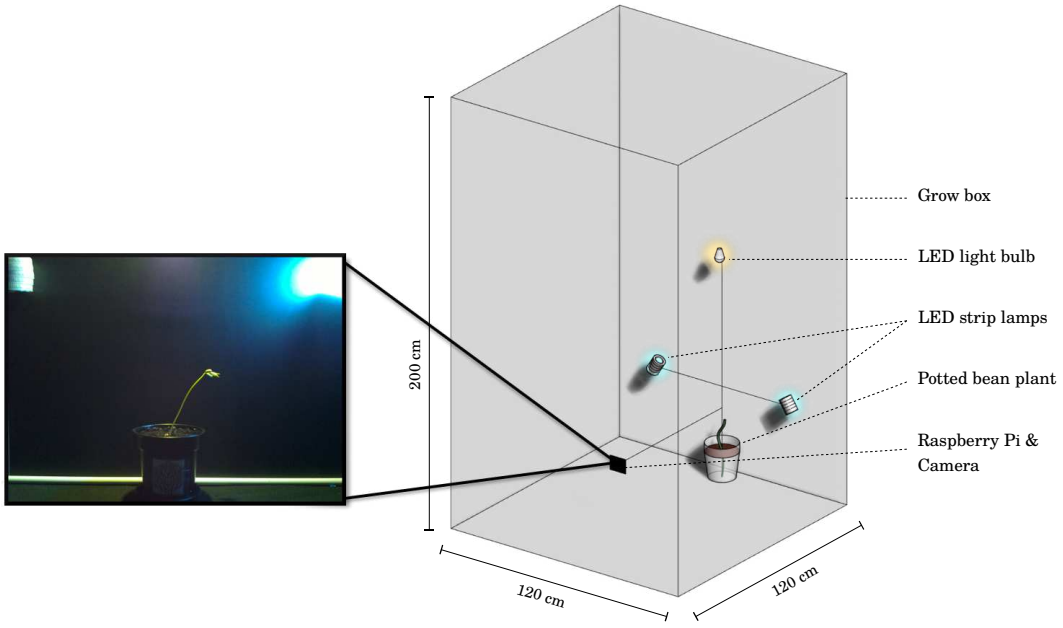


Figure 15: Bio-hybrid system setup. The setup is housed in a grow box, with the plant near the back wall and the camera near the front wall, facing the plant. A pair of light sources that provide stimuli are mounted above the plant on either side, right and left, and a flash for the camera hangs directly above the plant.

The preliminary experiments record the growth patterns of plants exposed to light sources from trivial manually-defined controllers, and a sampling method interprets the images into a raw data set of plant tip positions. Then, this data is used to build a purpose-specific model of tip-motion that enables simulation of tip trajectories in our setup under any given light source sequence. Finally, controllers are evolved in simulation using the *tip-motion model*, for the task of steering a plant tip to reach sequences of arbitrary user-defined targets. The method used for generation of user-defined targets is described, along with the two types of fitness functions used for evolution.

Bio-hybrid setup. The bio-hybrid system contains one biological plant that has a simple symbiotic relationship with centrally controlled robotic elements. The robot influences the plant by triggering directional light sources in its environment, and the plant influences the robot through on-board image sampling that detects plant dynamics.

As in the previous work [44], the plant in this setup is the common bean plant (*Phaseolus vulgaris L. var. nanus cf. Saxa*, a bush bean[3]), germinated in commercial soil intended for growing vegetables[4]. The bean is planted in a 1.5 liter pot, with a top diameter of 15 cm and soil level of 12 cm from the base.

The robotic portion of the system is subject to a centralized controller, and consists of the following elements: two NeoPixel LED strip lamps[5] for providing stimuli, a camera module[6]

[3] https://shop.nebelung.de/gemuesesamen/bohnen/buschbohnen-saxa.html
[4] FloraSelf Gemüse- und Tomatenerde ohne Torf (Floragard Vertriebs-GmbH)
[5] Adafruit NeoPixel RGB LED strips (https://www.adafruit.com/products/1506)
[6] Raspberry Pi camera module (https://www.raspberrypi.org/products/camera-module/)

to record the growth and motion of the plant by capturing photographs, an LED light-bulb[7] which provides flash for capturing photographs, and a Raspberry Pi[8] as a central control and processing unit. A NeoPixel strip contains 144 individually controllable integrated RGB LEDs[9], with peak-emission at wavelengths $\lambda_{\max}$ 630nm, 530nm, and 475nm respectively. Each LED consumes 0.24W when emitting white light at full power, giving 18 lumen. As we only trigger the two NeoPixel strips individually (never simultaneously), we can expect a total power consumption of up to approximately 35W.

Various scripts were developed to detect the plant tip, run the evolved artificial neural networks (ANNs), control the light sources, and regularly upload the captured photos and log files to a Network-attached storage device (NAS). The scripts run on the Raspberry Pi as background processes managed by systemd[10]. The ZeroMQ[11] library is used to allow the necessary communication among these processes (e.g., sending a flash-light request before capturing a photo).

The bio-hybrid system is contained in a commercial grow box of dimensions 120 cm ×120 cm ×200 cm in width, depth, and height (see Fig. 15). The standard grow box is modified, with the interior walls clad in matte black foam board to reduce light reflections and provide a smooth, consistent background that is in high contrast to the plant. The potted bean plant is positioned at the center of the back of the grow box, such that the plant's root-shoot transition (i.e., the location where the germinated bean protrudes from the soil) is at about 12 cm height, 8 cm from the back wall, and 60 cm from each side wall.

The camera module is positioned near the front wall of the grow box, at a height of 32 cm. It faces the plant, with the focal plane parallel to the back wall, as diagrammed in Fig. 15. This placement positions the camera approximately 74 cm from the plant and 82 cm from the back wall. The flash-light used when recording photographs hangs at a height of 80 cm from the base of the potted plant, and is centered over the pot. The two NeoPixel strips are coiled in cylindrical shapes to form two LED strip lamps, and affixed to the back wall of the grow box at a height 30 cm above the root-shoot transition and 35 cm to either side (see Fig. 15).

4.3.2. Model of plant tip behavior

As described in [18], the simple data-driven model of plant growth and motion, through which we project plant tip trajectories in simulation and inform the evolution of our bio-hybrid controllers, is constructed according to the method described in this section. Our purpose-specific *tip-motion model* is based on time-lapse photographs of preliminary plant growth experiments. These images are sampled to detect the xy-coordinates of the plant tip, building a data set of timestamped tip positions under one of two triggered light sources. We use this data to calculate the time-normalized tip-motion vectors of a subsequent time step according to the current bio-hybrid system configuration. These motion vectors are used, in conjunction with statistical functions incorporating the stochasticity appropriate to plant growth, to construct our model of plant tip dynamics.

Tip-motion model. We use the data from our preliminary experiments and image processing to create a simple *tip-motion model*. The model used in this work is an extension of our prior work [44]. In the previous approach, we defined the system configuration at time step t as $(\mathbf{x}, L, C)_t$, where $\mathbf{x}_t$ is the plant's tip position, L_t is the plant's length, and C_t is the lighting condition (Boolean value indicating whether the right light is on). The model simply predicted

[7] Philips LED bulb 8718696490860 (http://www.philips.co.uk/c-p/8718696490860/)
[8] Raspberry Pi 3 Model B (https://www.raspberrypi.org/products/raspberry-pi-3-model-b/)
[9] WS2812 integrated light source (https://cdn-shop.adafruit.com/datasheets/WS2812.pdf)
[10] Systemd is a system and service manager for Linux operating systems
[11] http://zeromq.org/

the next plant's tip position $\hat{\mathbf{x}}_{t+1}$, given the current system configuration $(\mathbf{x}, L, C)_t$. This was achieved by windowing into the recorded data during run time of the simulations and, from this window, calculating average position-changes that are corrected for consistent plant-lengths using standard trigonometry [44].

In this work, in order to greatly improve the speed of the plant trajectory simulations, our extension relies on an array of precomputed model statistics for each pixel location where a tip could be detected. During run time, the position of the detected tip is used to index into this array, retrieving the values needed for simulation. Because the model statistics are precalculated and therefore do not require short computing times, we use more sophisticated methods of aggregating the data. This allows the plant length parameter L_t to be removed from the model, simplifying the full description of the system configuration to $(\mathbf{x}, C)_t$.

In summary, we first aggregate the tip detection data into a four-column 2D array storing normalized and mirrored x, y, Δx, and Δy values from all six preliminary experiments. We then window into this new data to calculate the averages and standard deviations of the Δx and Δy values contained in the area around each pixel. We finally create two 3D arrays (one for the left light source, one for the right light source) that both have the same xy measurements of the experiment images (where the rows and columns correspond to those of an image), and have four layers in the third dimension, containing the mean Δx, mean Δy, standard deviation of Δx, and standard deviation of Δy. These two arrays comprise the *tip-motion model* and are used to compute the next tip position during simulation of growth and motion dynamics. Following is a more detailed description of the procedures.

Aggregating the tip data. Each tip position change from $\mathbf{x}_t$ to $\mathbf{x}_{t+1}$ occurs under the influence of Boolean light condition C_{t+1}. For each active light source, we first construct sets L (left light) and R (right light) containing tip positions $\mathbf{x}_t = (x_t, y_t)$,

$$L = \{\mathbf{x}_t | \quad C_{t+1} = 0\}, \quad R = \{\mathbf{x}_t | \quad C_{t+1} = 1\}. \tag{9}$$

We assume our setup to be symmetric and the plant to lack directional bias, therefore we mirror the data by transforming all $\Delta\mathbf{x}_t$ values associated with the left light source ($C_{t+1} = 0$) such that they provide additional data for the right light source. We calculate the tip position changes for all time steps t (excepting the final time step for each experiment) and time-normalize it by $\Delta\mathbf{x}_t = (\mathbf{x}_{t+1} - \mathbf{x}_t)/\Delta u_t$, where $\Delta u_t = u_{t+1} - u_t$ is the duration in minutes between every time step t. These vectors, along with positional data, are mirrored according to the width w of the processed image, and the data recorded in both light conditions is pooled together. In this way, all tip-motion data is aggregated into a single data set cast relative to the right light source. It is later reinterpreted to apply to both light sources, during the step of building the *tip-motion model*. Finally, the time-normalized and light-mirrored $x_i, y_i, \Delta x_i, \Delta y_i$ data for all experiments is pooled into a 4-column 2D array D_R, where

$$D_R = \{(x_i, y_i, \Delta x_i, \Delta y_i) | (x_i, y_i) \in R\} \cup \{(w - x_i, y_i, -\Delta x_i, \Delta y_i) | (w - x_i, y_i) \in L\}. \tag{10}$$

Building the tip-motion model. Our purpose-specific *tip-motion model* consists of two parts, each part for one of the two light conditions (left light source or right light source). The two parts are each structured as a 3D array, matching the x-axis and y-axis pixel count of the sampled images in two dimensions, and layering four sets of values in the third dimension. The first array of the model, for the right light source, is calculated from the $(x_i, y_i, \Delta x_i, \Delta y_i) \in D_R$ described above. The second array is later calculated from the first. We utilize inverse distance weighting (IDW) [35] in calculating the first array, to interpolate and smooth the data. The four

values calculated are the IDW averages of x-axis and y-axis tip-motion vectors, along with their respective standard deviations.

In contrast to the method in prior work [44], which windows into the aggregated tip-motion data using small rectangles (37.5 pixels × 75 pixels in width and height), here we window into the data using large circles of radius $r = 200$ pixels. Also, while the prior work [44] models tip-motion differently along the x-axis and y-axis, here we model them jointly, taking the IDW average and standard deviation of the windowed data for both axes.

To calculate the model for the right light source, we first window into the aggregated data D_R and construct array $W_{\mathbf{x}_p}$, which has dimensions matching the sampled images. For each position $\mathbf{x}_p = (x_p, y_p)$, the data points of D_R where $\mathbf{x_i} = (x_i, y_i)$ falls into window radius $r = 200$ is collected as

$$W_{\mathbf{x}_p} = \{(x_i, y_i, \Delta x_i, \Delta y_i) \in D_R \quad | \quad \|\mathbf{x}_i - \mathbf{x}_p\| \leq r\}. \tag{11}$$

This data is used to compute the IDW averages and standard deviations of Δx_i and Δy_i at each $\mathbf{x}_p$. A weight w_i is assigned, based on a simple IDW function with exponent two, to every data point $\mathbf{x_i} = (x_i, y_i)$ where $(x_i, y_i, \Delta x_i, \Delta y_i) \in W_{\mathbf{x}_p}$, such that

$$w_i = \begin{cases} 1/\|\mathbf{x}_p - \mathbf{x}_i\|^2 & \text{if } \|\mathbf{x}_p - \mathbf{x}_i\| \geq 12 \\ 1/12^2 & \text{else} \end{cases}. \tag{12}$$

If the distance is under 12 pixels in this operation, we fix it at 12 to avoid overweighting and division by zero, as our tip detection occurs at 1/8 resolution. The IDW averages for each position $\mathbf{x}_p = (x_p, y_p)$ are computed according to $(x_i, y_i, \Delta x_i, \Delta y_i) \in W_{\mathbf{x}_p}$ as

$$\mu(\Delta x_p) = \frac{1}{|W_{\mathbf{x}_p}|} \sum_{i=1}^{|W_{\mathbf{x}_p}|} w_i \Delta x_i, \tag{13}$$

$$\mu(\Delta y_p) = \frac{1}{|W_{\mathbf{x}_p}|} \sum_{i=1}^{|W_{\mathbf{x}_p}|} w_i \Delta y_i, \tag{14}$$

and the IDW standard deviations are computed as

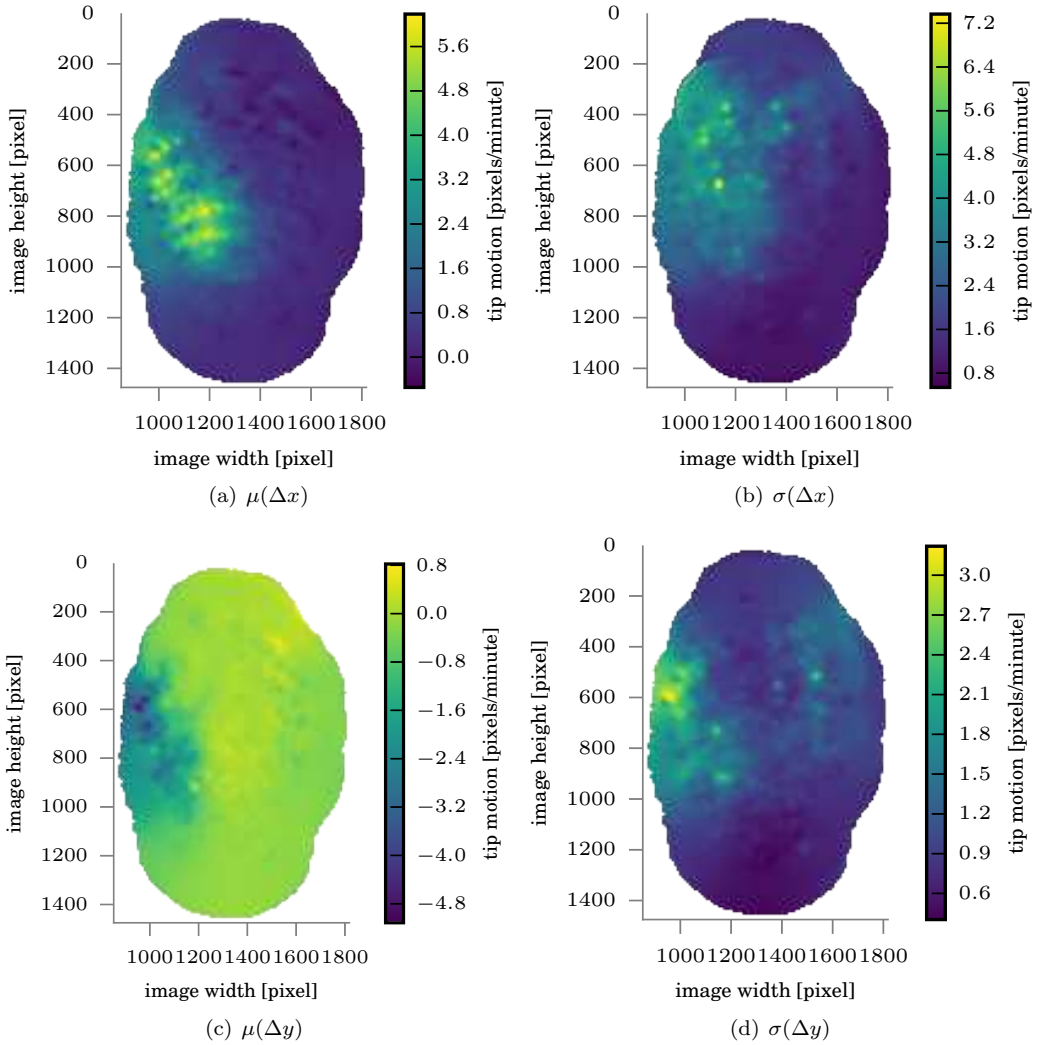
$$\sigma^2(\Delta x_p) = \frac{1}{|W_{\mathbf{x}_p}|} \sum_{i=1}^{|W_{\mathbf{x}_p}|} (w_i(\Delta x_i - \mu(\Delta x_p))^2), \tag{15}$$

$$\sigma^2(\Delta y_p) = \frac{1}{|W_{\mathbf{x}_p}|} \sum_{i=1}^{|W_{\mathbf{x}_p}|} (w_i(\Delta y_i - \mu(\Delta y_p))^2), \tag{16}$$

unless $|\mathbf{x}_i \in \mathbf{x}_p| < 30$ (i.e., there are fewer than 30 data points in the circular window around $\mathbf{x}_p$). In such cases, we discard the entries and cast that $\mathbf{x}_p$ position as empty, because there is not enough tip-motion data to calculate the model values.

A new 3D array M_R is constructed with two dimensions matching the sampled images, and with the values $\mu(\Delta x_p)$, $\mu(\Delta y_p)$, $\sigma(\Delta x_p)$, and $\sigma(\Delta y_p)$ for each $\mathbf{x}_p$ assigned to four layers in the 3rd dimension. This array M_R contains all the information of the model, but is cast under the right light source, as described above. To calculate the second array M_L of the model, we mirror M_R by flipping the four layers over the x-axis and by multiplying the $\mu(\Delta x_p)$ values by -1. This yields the final *tip-motion model* arrays, M_L and M_R.

The values contained in the *tip-motion model* array M_R (i.e., for the right light source) can be seen in Fig. 16. Figs. 16(a) and 16(c) represent the mean tip-motion vectors $\mu(\Delta x)$ and



(a) $\mu(\Delta x)$ (b) $\sigma(\Delta x)$

(c) $\mu(\Delta y)$ (d) $\sigma(\Delta y)$

Figure 16: The four layers of the *tip-motion model* array M_R, describing tip-motion when the right light source is triggered. The colormaps indicate the distribution of the IDW means μ and standard deviations σ of the Δx and Δy values at each xy position. The data is mirrored, time-normalized, smoothed, interpolated, and extrapolated according to the method described in Section 4.3.2. White patches indicate absence of data. Axes are given in image pixel coordinates. 37.5 pixels correspond to 1 cm in the camera focal plane occurring at the plant. (a) and (b) show the statistical description of the tip-motion on the x-axis, while (c) and (d) describe motion on the y-axis. Because the y-axis origin is placed at the top of the image (above the plant), negative values in colormap (c) indicate the plant tip's upward motion. In (a), positive values indicate motion towards the right.

$\mu(\Delta y)$, Fig. 16(b) and Fig. 16(d) show the standard deviations $\sigma(\Delta x)$ and $\sigma(\Delta y)$. The standard deviation values are usually larger than their respective means, showing the large variance in the dataset. In Fig. 16(c), there is a general trend of growth upwards (negative values on the colorscale indicate upward motion, as $y = 0$ is the uppermost pixel) that is especially pronounced on the far side from the light source. Towards the top-right corner of the data is a concentration of motion downwards. Both of these features in the data reveal the plant stem bending towards the light source. Fig. 16(a) indicates that, when the tip is farther from the light source, it typically moves more quickly towards it. The features visible in these colormaps of M_R would be mirrored horizontally in colormaps of M_L. These *tip-motion model* arrays, M_L and M_R, allow for significantly improved and sped up plant simulations, as compared to prior work [44].

Simulation of tip behavior. We begin simulations of plant tip trajectories at a manually fixed origin (x_0, y_0). The origin x_0 is placed at the horizontal center of the image ($x_0 = 1296$ pixels), and y_0 is placed at the lowest point in the photographs that occurs higher than the edge of the pot in all experiments ($y_0 = 1250$ pixels, where $y = 0$ occurs at the uppermost row of the image, above the plant). We use this (x_0, y_0) origin as our first simulated plant tip position $\mathbf{x}_t = (x_t, y_t)$ in each simulation run. At each time step of the simulation, we index into the appropriate array M_L or M_R depending on the light condition output by the controller, and according to the current simulated tip position $\mathbf{x}_t$ retrieve the stored $\mu(\Delta x_i)$, $\mu(\Delta y_i)$, $\sigma(\Delta x_i)$, and $\sigma(\Delta y_i)$. We compute the next tip position $\mathbf{x}_{t+1}$ by drawing from random distributions according to the retrieved values, such that

$$x_{t+1} = x_t + \mathcal{N}(\mu(\Delta x_i), \sigma(\Delta x_i)), \tag{17}$$

$$y_{t+1} = y_t + \mathcal{N}(\mu(\Delta y_i), \sigma(\Delta y_i)). \tag{18}$$

4.3.3. Controller setup for user-defined targets

As described in [18], our controller is an ANN with four inputs. The inputs are the xy coordinates of the current plant tip position and the current target position at each time step. These time steps correspond with five minute discrete intervals in reality. The network has only one output, which is a binary decision indicating whether the left or right light source will be triggered in the next time step. When the controller runs in reality gap experiments, an image of the plant is captured and processed at each time step, acquiring the current position of the real plant tip. When the controller runs in simulation, the *tip-motion model* is used to project the current tip position from the previous.

Definition of the task. In our previous work [44], the controller performed the task of steering the plant tip serially towards three specific, unchanging targets in xy space. Here, we extend the work by generalizing the controller, such that the next arbitrary xy target is provided as an input.

This enhancement allows the controller to direct a wide range of growth patterns for various applications (e.g., plant shaping, braiding). A new target input to the controller is triggered once the current target $\mathbf{x}_i^*$ is reached by the plant tip $\mathbf{x}_t$, such that

$$|x_t - x_i^*| \leq 12 \vee |y_t - y_i^*| \leq 12 \vee |y_i^* - y_t| \geq 40. \tag{19}$$

Using this approach of analyzing target proximity separately along the two axes eliminates the speed problem posed by calculation of Euclidean distance, at a time-critical step. When triggered, a new target is generated according to the method described below.

Target generation. The task of the evolved controllers is to steer the plant tip to arbitrary targets that can change at run time. Targets would be user-defined in an application, hence, for benchmarking we automatically generate a varying number of arbitrary targets for every individual controller and evaluate its behavior according to those target sets. Besides the objective of targets being reachable by the plant in principle, we have two competing priorities in our generation method. On the one hand, because we evaluate each controller with multiple simulation runs (each with varying numbers of targets), the target sets should have comparable statistical properties, such that the evaluation of controller performance is not biased by differences in targets. On the other hand, the generated targets should be diverse enough to ensure that the evolved controllers will be general. We approach these objectives by using a method similar to our simulation of tip-motion, but instead of drawing random values from a normal distribution parameterized from the *tip-motion model* array, we select values corresponding to a given cumulative probability at each pixel. As such, we employ a data-driven probabilistic target generation method, described below. A new target is generated once a previous target is reached. The position of the new target is defined based on a given probability for reachability of the point and the desired relative distance along the trajectory from the tip position to the edge of data in the array.

The function for target generation (see Algorithm 1) receives the position of the plant tip at the time of target generation (tip_x, tip_y), and the *tip-motion model* arrays M_L for left light source M_R and right light source. The function also uses a set of parameters that can be changed by the user according to the desired level of reachability of the targets, p_{reach}, the level of desire for switching direction when moving from one target to the next, p_{switch}, and the relative distance d_{traverse} along the projected trajectory where we fix the target, the maximum y position $y_{\max}$ of any target in the system, and lastly, the direction r_{last} in which the prior target was generated. In the implementation used here, we draw uniform random values for p_{reach} from the half-open interval $[0.58,\ 0.72)$ and for p_{traverse} from $[0.5,\ 0.7)$ every time a new target is generated. The value of p_{switch} is fixed at 0.75.

In order to generate a trajectory in a certain direction, we use the $\mu(\Delta y_i)$ and $\sigma(\Delta y_i)$ from the *tip-motion model* array of the respective light source C (left light, $C = 0$, right light, $C = 1$). Based on that, we calculate a trajectory of positions that are reachable with the desired probability p_{reach}. The set of positions are traversed in order to choose a target with the desired distance from the previous one based on d_{traverse}. Starting from the last target, we iteratively generate a set of potential target positions $\mathbf{x}_p$ where each point is adjacent to the previous one, such that

$$x_{p+1} = \begin{cases} x_p - 1, \text{ if } C = 0 \\ x_p + 1, \text{ else} \end{cases}, \tag{20}$$

$$y_{p+1} = y_p + \mu(\Delta y_i) - z\sigma(\Delta y_i). \tag{21}$$

where z is the multiplier for the standard deviation $\sigma(\Delta y_i)$ associated to position (x_p, y_p). The probit[12] function Φ^{-1} is used to convert the desired cumulative probability p_{reach} into the corresponding value $z = \Phi^{-1}(p_{\text{reach}})$ from a standard normally distributed variable, that we can then use to get the according value for any normal distribution (by multiplying with its σ).

The resulting list of pixel coordinates is defined as the trajectory from which a target will be selected (cast as border trajectory b in Algorithm 2). The target is chosen by traversing this series of pixels to reach the distance d_{traverse}, which gives the relative distance from the previous target (see Algorithms 1 and 2 for further details).

[12]The Scipy Python library is used to apply the probit function; that is the quantile function associated with the standard normal distribution; it is the inverse of the cumulative distribution function (CDF).

ALGORITHM 1: Target Generation

Data: tip position (tip_x, tip_y), *tip-motion model* arrays M_L, M_R, given probability for reaching the target p_{reach}, given probability to switch direction of target generation (left or right of tip) p_{switch}, the given desired relative distance d_{traverse} between the old and new target positions, the Boolean r_{last} that is true when the previous target was generated towards the right, the maximum height for target $y_{\max}$

Result: target position $(targ_x, targ_y)$, Boolean c_r (true if the new target was generated towards the right)

$r \leftarrow$ uniform random number in $[0, 1)$;
$b \leftarrow$ empty list of border pixel coordinates;

if $r < p_{\text{switch}}$*;* `/* i.e., we switch direction */`
then
  if *not* r_{last}*;* `/* i.e., last target was generated towards the left */`
  then
    $c_r \leftarrow True$;
    $M_C \leftarrow M_R$; `/* pick the array for right light */`
  else
    $c_r \leftarrow False$;
    $M_C \leftarrow M_L$;
  end
else
  if r_{last} **then**
    $c_r \leftarrow True$;
    $M_C \leftarrow M_R$;
  else
    $c_r \leftarrow False$;
    $M_C \leftarrow M_L$;
  end
end

$b \leftarrow getTrajectory(tip_x, tip_y, M_C, p_{\text{reach}})$; `/* call function getTrajectory() */`

$b_{\text{len}} \leftarrow$ length of b; `/* number of tuples` (b_x, b_y) `in list */`

if $b_{\text{len}} > 0$*;* `/* i.e., plant tip was at valid entry of model array */`
then
  $b_{\text{index}} \leftarrow floor(b_{\text{len}} \cdot d_{\text{traverse}})$; `/* index of target tuple according to` $d_{\texttt{traverse}}$ `*/`
  $(targ_x, targ_y) \leftarrow b[b_{\text{index}}]$;
  if $targ_y < y_{max}$*;* `/* if target is higher than allowed */`
  then
    $(targ_x, targ_y) \leftarrow$ the first tuple from b that is lower than $y_{\max}$;
  end
else
  generate a random target in the central data area (with $targ_y$ lying above the tip_y and below $y_{\max}$);
end

return tuple $(targ_x, targ_y)$ and Boolean c_r

ALGORITHM 2: Function getTrajectory()

Data: tip position (tip_x, tip_y); $\mu(\Delta y_i)$ and $\sigma(\Delta y_i)$ from the *tip-motion model* array M_C for light condition C (Boolean value, equal to 1 for right light); the probability for reaching a target p_{reach}

Result: list of tuples (b_x, b_y) border trajectory b

$z \leftarrow \Phi^{-1}(p_{\text{reach}})$; `/* probit function */`
$b \leftarrow$ empty list of border pixel coordinates;

if *C is True;* `/* i.e., right light is lit */`
then
  $m \leftarrow 1$; `/* next target right */`
else
  $m \leftarrow -1$; `/* next target left */`
end

$x_p \leftarrow tip_x$;
$y_p \leftarrow tip_y$;
while *not ($\mu(\Delta y_i) ==$ 'NaN');* `/* i.e., we have data for that location */`
do
  append the tuple (x_p, y_p) to the list b;

  $x_p \leftarrow x_p + m$;
  $y_p \leftarrow y_p + \mu(\Delta y_i) - z \cdot \sigma(\Delta y_i)$; `/* using the quantile on σ */`

  $[\mu(\Delta y_i), \sigma(\Delta y_i)] \leftarrow$ aisle of M_c at row $m = round(y_p)$ and column $n = round(x_p)$;
end

return the list of tuples b constituting all points approximately reachable with probability p_{reach} under the given light condition C

Table 1: Used NEAT parameters.

Parameter	**Value**	**Parameter**	**Value**
PopulationSize	50	*CrossoverRate*	0.5
DynamicCompatibility	True	*MutateWeightsProb*	0.9
YoungAgeTreshold	15	*YoungAgeFitnessBoost*	1.0
OverallMutationRate	0.5	*WeightReplacementMax*	5.0
MinSpecies	5	*WeightMutationRate*	0.75
MaxSpecies	25	*Elitism*	0.1
SurvivalRate	0.6	*MutateAddNeuronProb*	0.04

Evolutionary approach. We evolve ANN controllers using the portable Python library MultiNEAT [5], which is based on NeuroEvolution of Augmenting Topologies (NEAT) [39]. NEAT is an efficient evolutionary algorithm that begins with a random population of ANNs with minimal structure (i.e., no hidden layers), then applies complexifying methods to modify the weights and the network structure. We use here the set of NEAT parameters in Table 1, based on the parameters that have shown successful performance in our previous work [44].

At each time step t, the xy coordinates of the current plant tip position $\mathbf{x}_t = (x_t, y_t)$ and current target position $\mathbf{x}_i^* = (x_i^*, y_i^*)$ are input to the ANN. The ANN then outputs the binary light source state C_t, determining whether the left light source ($C_t = 0$) or the right light source ($C_t = 1$) is activated. The current system configuration $(\mathbf{x}, C)_t$ influences the plant's behavior (growth and motion) during that time step. Therefore, in the case of simulation, the *tip-motion model* is used to project the next tip position $\mathbf{x}_{t+1}$ from the current system configuration. In reality gap experiments, an image of the plant is captured after the time step is complete, and the processing method described in Section 4.2.1 is used to detect the plant's new tip position. This process is repeated at every time step until the tip's y_t value is equivalent to roughly 20 cm in height. This allows the plant enough growth space to reach between two and six arbitrary targets generated by the method described above. It also allows the reality gap experiments to be performed in a relatively short period of time (approximately 72 hours each), such that the overhead is manageable for an engineering task. In order to evolve ANN controllers to steer the plant tip to generalized targets, we define two alternative fitness functions, F_1 (a behavioral fitness function) and F_2 (an aggregate fitness function), according to the classification in [31].

First, we define the behavioral fitness function F_1. Since the motion control acts mainly along the x-axis while the change along the y-axis is mostly due to growth, we measure the distances traversed towards every new target from the previous target along the x-axis. For that, for every target $i \in \{1, 2, ..., n\}$ we define the instant rewards $r(t)$, based on the number N of total targets and the distance Δx_t traversed in every time step t:

$$\Delta x_t = x_t - x_{t-1}, \quad t \in \left\{ t \mid y_{i-1}^* \leq y_t < y_i^* \right\}, \tag{22}$$

$$r(t) = \begin{cases} \Delta x_t, & \text{if } x_t < x_i^* \\ -\Delta x_t, & \text{if } x_t > x_i^* \\ |\Delta x_t|, & \text{if } x_t = x_i^* \end{cases}, \tag{23}$$

where (x_t, y_t) is the position of the tip at time step t and (x_i^*, y_i^*) is the position of the target i. (x_0^*, y_0^*) is the starting position of the tip. The sum of the rewards for each target is defined as

$$R_i = \sum_t r(t), \quad t \in \left\{ t \mid y_{i-1}^* \leq y_t < y_i^* \right\}. \tag{24}$$

The controller is rewarded R_i when the tip transitions between the old and the new targets. If the tip starts out of this region we make a correction in the value of R_i by decreasing it by $x_i^* - x_t$ where x_t is the starting point of the tip for the corresponding target. The reason for this correction is to prevent solutions where the tip compensates for missing a target by getting extra reward for the next target, through starting its motion towards the new target at a farther distance. The behavioral fitness F_1 is then computed as follows:

$$F_1 = \frac{\sum_{i=1}^{N} R_i}{\sum_{i=1}^{N} R_i^{\max}}, \tag{25}$$

where $R_i^{\max} = |x_i^* - x_{i-1}^*|$ is the maximal reward that a controller can theoretically achieve for every target i, and N is the number of targets. During every epoch of the artificial evolution, we evaluate each controller in 15 independent plant growth simulations (with distinct generated targets) and select the minimum fitness as the controller's final fitness.

F_1 is a behavioral fitness function; namely, it has prior knowledge about possible useful behavior leading to a potential solution. It uses this knowledge to continuously reward/punish the controller according to its behavior.

To define the second fitness function F_2, we use the Euclidean distance between the current plant tip position (x_t, y_t) and the considered target i, where

$$\mathrm{dist}_i(t) = \sqrt{(x_i^* - x_t)^2 + (y_i^* - y_t)^2}. \tag{26}$$

A controller receives a reward of $R = 1$ if it reaches the vicinity r of the target, at any time step $t \in T$ when the plant tip is positioned between the heights of the current and former target, such that

$$T_i = \left\{ t \mid y_{i-1}^* \leq y_t < y_i^* \right\}, \tag{27}$$

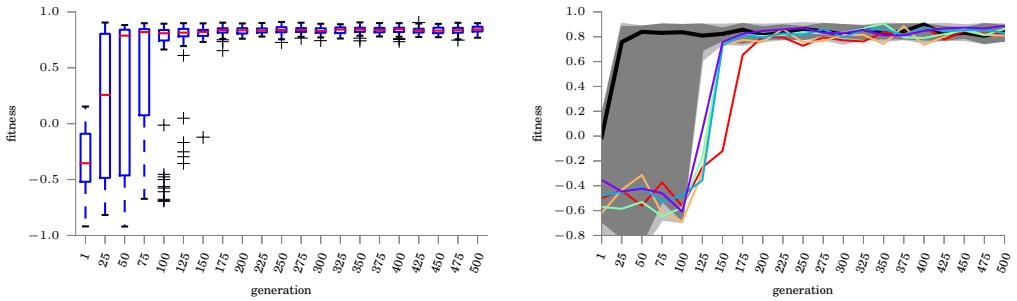
$$R_i = \begin{cases} 1, & \text{if } \exists t \in T_i, \mathrm{dist}_i(t) \leq r \\ 0, & \text{else} \end{cases}. \tag{28}$$

F_2 is then defined according to the reward values, as

$$F_2 = \frac{1}{N} \sum_{i=1}^{N} R_i \ . \tag{29}$$

Similarly to F_1, for the controllers evolved using F_2, we evaluate each controller according to 15 independent target sets and plant growth simulations. However, instead of selecting the minimum, for F_2 we calculate the average fitness of the 15 runs, setting that average as the controller's final fitness.

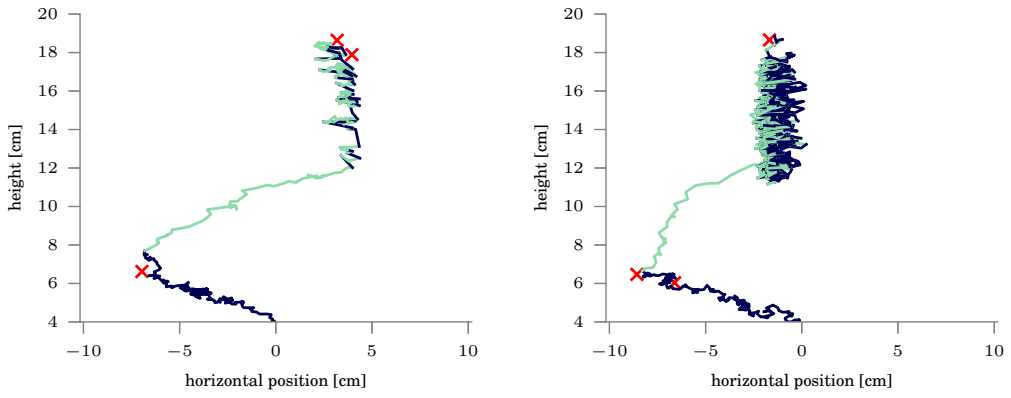
In contrast to F_1, the F_2 is an aggregate fitness function, therefore the completion of the task is the only metric for evaluation, regardless of the procedure leading to the solution. Here, the controller is only rewarded when the tip approaches a target (see eq. 28). Therefore, F_2 is less complex than the behavioral F_1, but the lack of guidance in F_2 could lead to bootstrapping problems, slowing down the evolutionary process [10]. We do not encounter this potential slowness, by virtue of the array approach to modeling and simulating growth (see Section 4.3.2), which greatly speeds our evolutionary process. This allows us to investigate the aggregate fitness function approach seen in F_2.



(a) Fitness of best controller per generation.

(b) Functional boxplot.

Figure 17: Evolution of controllers selected by fitness function F_1. Shown here are 50 independent runs. (a) Boxplots give the first quartile, the median, and the third quartile, while the whiskers are 1.5 times the interquartile distance added to the box. Outliers are shown separately. (b) Because the generation-fitnesses of a single run are independent of each other, we can use functional boxplots. The figure shows the "median-function" in black, while the "quartile" and "whisker functions" color areas dark and light gray respectively. The "whisker functions" are 3 times the interquartile distance added. A function is considered an outlier and drawn separately, if it is outside the range at any single generation.



(a) Evolved controller using F_1, success = 92.3%

(b) Evolved controller using F_1, success = 91.0%

Figure 18: Trajectories of the simulated plant tip for two successful controllers using F_1

4.3.4. Results of evolved controllers

As described in [18], first, we report the results of evolving controllers using the *tip-motion model* (see Sec. 4.3.2) in simulation. We test our previously mentioned two fitness functions (F_1 and F_2) in two sets of 50 evolutionary runs, 500 generations each. Second, we report the performance of the evolved controllers in plant experiments (i.e., discuss the existence of reality gap).

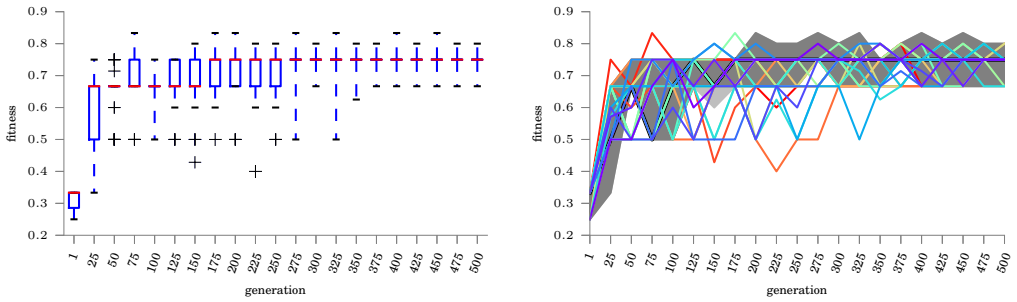
Evolution of controllers in simulation Compared to [44], the optimized procedure allowed, but also required, the stricter evaluation scheme of testing each individual controller on 15 simulation runs. The current model includes substantially more (and locally inhomogeneous) stochasticity, better reflecting the plants' behavior. First, we show the results for the behavioral fitness function F_1. Fig. 17 shows boxplots and functional boxplots of 50 independent runs that each contain 500 generations. We can clearly see that, unlike in the previous work [44], we can much better guarantee that the NEAT process finds a solution to this more complex problem. By generation 200, we have reached convergence, while the majority of populations evolved a solution within the first 50 generations. Counterintuitively, feeding the ANN the current target as an additional input parameter makes this task much easier, as the network can learn the correlation between the input and output parameters. Fig. 18(a) shows the worst performance of 15 plant simulations of the controller we selected to guide the plants across the reality gap. We chose it because it had the highest fitness of all controllers, even though we are aware that there is a lot of stochasticity involved. The strategy is straightforward: trigger the light source that leaves the target in between the light source and the plant tip. This leads to the plant being guided below the target as fast as possible, and then kept there (by alternating the lights) until it grows to reach the target. This behavior is particularly pronounced in Fig. 18(b), where the tip sometimes moved down, giving a thicket of trajectory very similar to one of our reality-gap experiments[13].

Second, we show the results for the aggregate fitness function F_2. The results from 50 independent runs are shown in Fig. 19. Following the median in Fig. 19(a), the fitness increases steadily and saturation is achieved after 225 generations. Notice that the median shows step-wise increase/decrease behavior which reflects the properties of an aggregate fitness function (explicit reward when a target is achieved).

Fig. 22 shows the plant tip trajectories resulting from two successful controllers. The strategy evolved is the same as that employed by controllers selected with F_1 (see Fig. 18). In Fig. 20(a), five targets were generated and the controller could score 85.7% success. Here, the controller could steer the plant tip towards four targets successfully, but fails to approach the last target due to stochasticity in simulated plant behavior. In Fig. 20(b), another four targets were generated and the controller successfully guided the simulated tip towards all of them, giving a score of 100%.

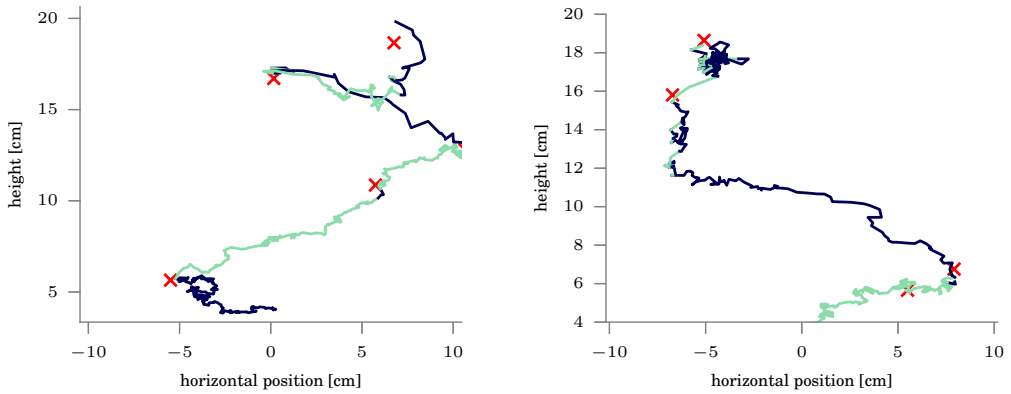
Performance of controllers in plant experiments In a final set of plant experiments, one of our controllers (see Fig. 18(a)), that was successfully evolved in simulation, is tested in reality with actual bean plants. It is one of the controllers evolved based on the behavioral fitness function F_1. This is a typical attempt to investigate the reality gap problem [23]. In our previous work [44], we confirmed the possibility to bridge the reality gap for a task with three predefined target points ($\mathbf{x}_1^* = (3, 6)$, $\mathbf{x}_2^* = (-5, 9)$ and $\mathbf{x}_3^* = (-1, 13.5)$). Here, we test the performance of our evolved controller first in a similar scenario (fixed targets experiments). Second, we extend our reality gap investigation by including plant experiments with dynamically generated targets (dynamic-targets experiments) as described in Sec. 4.3.3. One of these plant

[13] Find a video of a fixed-targets and a dynamic-targets experiment at: https://vimeo.com/205469308



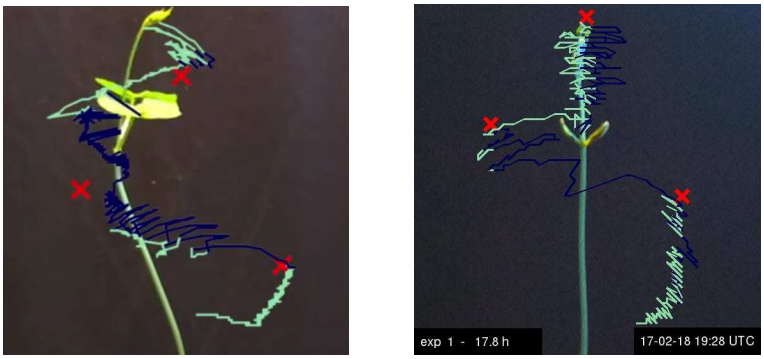
(a) Boxplot of best fitness per generation. (b) Functional boxplot of best fitness per generation.

Figure 19: Performance of the evolutionary process over generations for 50 evolutionary runs.



(a) Evolved controller using F_2, success = 85.7% (b) Evolved controller using F_2, success = 100%

Figure 20: Trajectories of the simulated plant tip for two successful controllers using F_2



(a) Trajectories from previous work, fitness = 72.6%

(b) Trajectories from current work, fitness = 95.17%

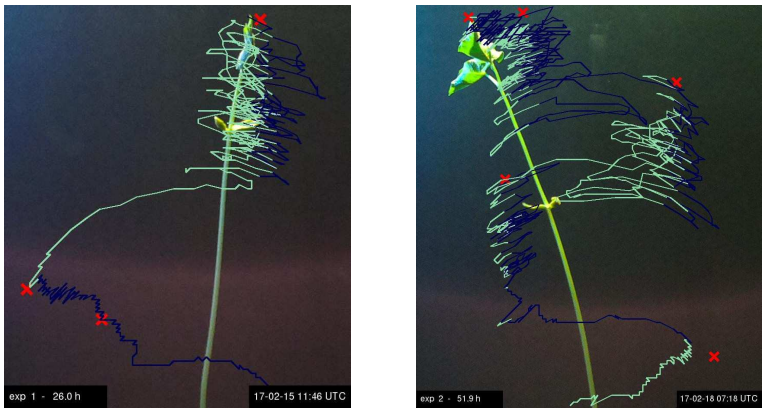
Figure 21: Trajectories of the detected plant tips in fixed targets experiments.

growth experiments takes up to 72 hours, therefore, we parallelized our experiments and tested the controller concurrently in two separate bio-hybrid experiment setups.

On the one hand, the fixed-targets experiment was repeated three times, scoring a fitness of 95.17% on average. Hence, we observe an average 28.7% increase in performance in comparison to our previous work. In Fig. 21, we show the trajectories of the guided plant tip from our previous work (Fig. 21(a)) side by side with the trajectories from our current work which scored a fitness of 95.17% (Fig. 21(b)). On the other hand, the dynamic-targets experiment was repeated two times, scoring a fitness of 91.25% in average. In Figs. 22(a) and (b) we show the trajectories of the guided plant tip from the two dynamic-targets experiments scoring fitness values of 92.6% and 89.9% respectively. In comparison to the experiments in simulation, we notice similar behaviors of the actual plant steered by the controller. The controller makes the tip of the plant approach every target precisely from below ($x_t = x_i^*$). Then it maintains the horizontal coordinate x_t during the plant's growth by alternating between the two light sources until the target is reached. This generates a series of back and forth movements (e.g., in Fig. 22(b) between the second and third target[14]. When the target is finally approached and switched, the controller rotates the plant appropriately to the opposite side (where required) beneath the next target.

Both the superficial observation of the plant behavior, that is, the effects of the control and the measured fitness in the plant experiments indicate that the controllers transferred successfully from simulations to reality without being changed. Hence, we have successfully bridged the reality gap in this specific setup. This is a very encouraging result because it demonstrates possible future pathways for this research. While the plant experiments are slow and expensive, our modeling approach allows to quickly evolve controllers with high transferability. Whether this feature scales up to scenarios of higher complexity needs to be shown.

[14]Find a video of fixed-targets and dynamic-targets experiments, at: https://vimeo.com/205469308



(a) Dynamic-targets experiment #1, fitness = 92.6%

(b) Dynamic-targets experiment #2, fitness = 89.9%

Figure 22: Trajectories of the detected plant tips in dynamic-targets experiments.

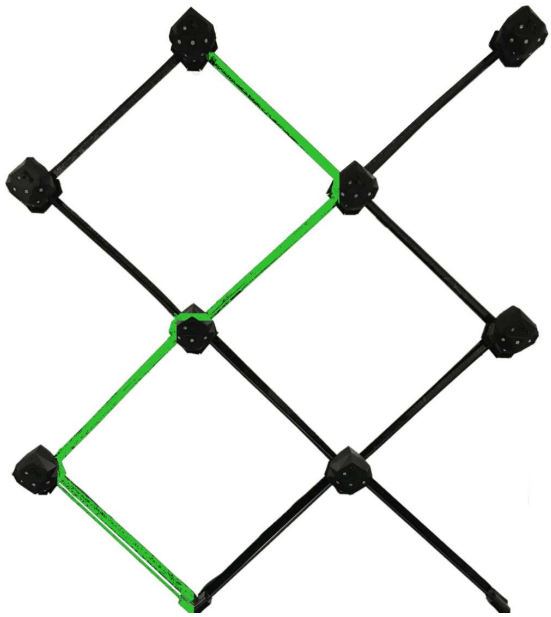


Figure 23: Desired user-defined Z-shaped pattern.

4.3.5. Distributed algorithm in the PBDW

Finally, we describe the control algorithm used for the PBDW experiment. More details about the PBDW setup are presented in deliverable *D1.2 Evaluation of mechatronics prototype of the robotic symbiont including supporting software.*

As described in [45], we define two states for the robotic nodes: *guiding*, the node emits only blue light to attract the plants and detects the proximity of approaching plant tips (proximity sensors are on); and *feeding*, the node emits only red light to support the plants' photosynthesis (proximity sensors are off, no plant tip detection). Then, we investigate the ability of our distributed and decentralized robot system to grow plants in predefined shapes. Every robotic node in the system knows its location in the diagrid map and the required final pattern (z-shaped pattern, see Fig. 23). Only one node is allowed to be in the *guiding* state at any given time. When the experiment starts, one node in the first level sets its state to *guiding* according to the predefined pattern. Once the node detects an approaching plant, it notifies its direct neighbors and switches its state to *feeding*. According to the pattern, the relevant node in the next level will switch its state to *guiding*. This process is repeated until the node in the final level detects an approaching plant.

5 Vascular morphogenesis as a model of self-organized growth in a bio-hybrid system

5.1 Introduction

Growth is a ubiquitous phenomenon in nature. Living organisms grow and develop their morphologies based on their genetic information, conditions of their environment, and the rules of physics and chemistry governing the dynamics of their world [17]. Morphology of biological systems develops over time in a process of growth changing their body and nervous system.

Here we are interested in the growth of the *flora robotica* bio-hybrids. The algorithm that is introduced here concerns the growth of the robotic side of the bio-hybrid in response to changes in the environment including the plant side and humans. The idea is to self-organize the growth of the structure in a distributed and balanced way. The distributed morphogenesis control algorithm runs independently in every module of the robot. The local controllers self-organize the process of growth based on local interactions with each other and with their environment, deciding how the growth should continue such that the global objectives are met.

Our morphogenesis approach is based on inspiration from the mechanisms of branching and branch competition in plants. Various branches of the same plant act as agents exploring their local environments and finding preferable regions and resources (e.g., light). Plants grow and extend their branches and develop new ones. Different branches compete for common global resources (e.g., water) and winning is biased towards the branches that find better local resources [16]. The information collected by the branches are used to find preferable regions of the environment leading to new shapes of the organism that facilitates access to more resources in a cascading way. In a more abstract view, branches are seen as members of a swarm competing for limited shared resources and it leads to new formations of the swarm that result in more overall resources for the system as a whole. Water, nutrients, and sugars are transported to different parts of a plant through the plant's vascular system. Evidence suggests that the vascular system not only transports materials needed for survival of the plant, but it is also a long-range communication channel that enables the plant to adapt to changes [32].

There are several biological models of plants, for example, modelling transportation of materials via vascular systems and incorporating it into growth models [50]. While the models developed in the field of plant science are mostly complex, focusing on details of particular plants, here we are interested in a simple decentralized model that captures the general concepts of growth such that it can be easily implemented in a limited robotic system. We are inspired from development of vascular systems in plants and their effects in the growth of different competing branches in an abstract level. We propose a novel controller called "Vascular Morphogenesis Controller (VMC)" for guiding the morphology of structures based on competition and resource balancing. The method differentiates between the information level (algorithmic logics of growth) and physical level (actual structure) in order to keep generality and broad scope of usability for the morphogenesis algorithm. The actual morphology of a structure being developed by this method is the result of parameters of the algorithm, and the resource sharing and competition mechanisms, as well as the physical realization of the system and their interactions with the environment on both informational and physical levels which leads to diverse possibilities of dynamic forms.

The algorithm is described in this chapter in the following sections. For the physical realization of a growing system, we have used a simulated modular robotic structure based on a physical modular robot and we have used it for evolving and developing various example structures in different conditions. The simulation results are reported in deliverable *D3.1 Representations and design rules*.

5.2 Related work

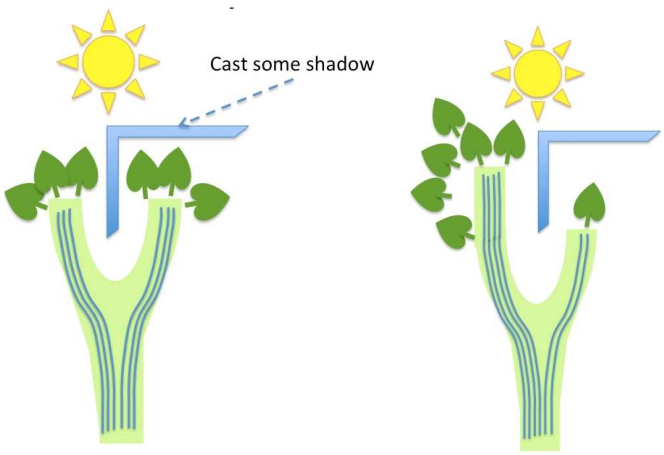
In biological studies, evolutionary developmental biology (EvoDevo) [22, 46] is the approach towards investigation of self-organized processes of growth by looking into embryogenetic development and differentiation of cells. Growth and morphogenesis is also a topic of interest in artificial and robotic systems, in particular in evolutionary robotics [1] and modular robotics [30]. In such artificial systems, as in their natural counterparts, patterns and structures are a result of self-organization. The different components interact with each other and with the environment leading to high diversity and adaptivity [17, 3, 48]. Self-organizing morphogenesis is usually implemented by using indirect encodings where the encoded parameters are subject to evolutionary algorithms for optimization. The body of the robot or the controller structure develop over time based on a set of parameters encoded in the genotype, a set of rules determining how those parameters drive the development, and the inputs to the system resulting from the interactions between the system and its environment (e.g., [36, 14, 49]).

While most of the work in the context of robotic morphogenesis is done in simulation, there are also a number of attempts towards real physical hardware [15, 2]. Many of the robotic approaches are strongly inspired from the concepts of real cell development, for example, by implementing variants of gene regulatory networks and the concepts of cell division, migration and death [8]. Other methods (e.g., [20, 36]) use the more abstract generative encodings, such as different variances of L-systems [27]. Generative encodings start with a basic unit (a seed) and a set of context-free developmental rules driving the development of the morphology. In [43] the L-system is extended by adding a swarm of interacting agents and a swarm grammar defining their dynamics. Another swarm-based morphogenesis model is the embryomorphic model [12] that implements the concepts of gradient diffusion for positional information, gene regulatory networks for transformation of cell types and cell division. The model draws a separation between the physical level and the information (logical) level of the growing system in order to keep the generality. In [24] a morphogenesis system based on Cellular Automata (CA) is defined with different types of cells including transport, barrier, and normal cells where the transport cells stay connected with each other in order to serve the normal cells during growth. Several different morphogenesis approaches are reviewed in [13]. The morphogenesis methods of structure development have also been applied to the field of computer networks. For example, in [38], a self-organized peer-to-peer overlay network develops in a computer network by using an algorithm inspired from fungal growth that maintains communication between nodes and demonstrates robustness against network failures.

5.3 Inspired by plants: competition for vessels

Vascular strands transport water and minerals from roots and sugars produced at the leaves to all over a plant. The vascular structure of a plant is dynamic as the vessels in different branches are reinforced or degenerated over time based on their status. Research in plant physiology [42, 34] suggests that different branches of a plant compete for more vessels. For example, experiments [34] with two-shoot pea seedlings, demonstrate that casting shadows on previously equally-placed shoots, for example, with equal access to light, causes the decrease and finally stopping of the shaded shoot while the other shoot grows faster and becomes dominant (see Fig. 24). This dominance can be reversed by restraining the dominant shoot that is located in light.

A hormone called auxin is produced at the tips of a plant and flows back via plant vessels towards the roots. Auxin production is influenced by the local conditions of the tips (e.g., access to light). One of the effects of auxin is to make the bundles of unspecialized stem cells, called cambium cells, which are located near the vascular tissues to transform into vessels. This is



(a) (b)

Figure 24: If one of the two equal branches (a) is in a comparatively preferable environmental condition (e.g., more light), it develops more vascular tissues and grows more vigorously (b).

especially interesting considering the fact that limited common resources (e.g., water) need to be distributed between different branches of a plant via their vessels. More flow of auxin at a branch, leads to more vessels, and thus more transportation of the common resources to the branch. More resources mean more growth and perhaps positioning of the branch's tip at even better locations generating a positive feedback loop. On the other hand, as common resources are limited for a plant, the distribution of the limited resource between all branches creates a negative feedback that pushes towards homeostasis of the plant.

5.4 The algorithm: Vascular Morphogenesis Controller

By taking inspiration from the competition between branches of a plant for more vessels and consequently more resources, the Vascular Morphogenesis Controller (VMC) algorithm is designed. The idea is to let the growable parts of a structure compete for a common resource of growth via a network of pathways that are dynamic based on interactions and local conditions of the branches. For that, VMC is defined as an acyclic directed graph (tree) that is distributed over the physical growing structure (see Fig. 25). Initially, the graph consists of a root node and a number of potentially growable nodes attached to the root making the initial leaves of the graph. Growth can only happen at the leaves of the graph. Each leaf assesses local conditions and accordingly produces a value, namely Successin (S) in analogy to Auxin in plants. The Successin produced at the leaves flows back to the root and on its way, it regulates the vascular pathways of the resource (V) – in analogy to plant vessels. The pathways are then used to distribute a limited resource (R) between the leaves. The limited resource is initiated at the root and flows towards the leaves being split at every branching point according to the thickness of the vascular pathways. The leaves that gain more share of the resource are more motivated to grow. When a leaf grows, new leaves are generated as its children, and hence the old leaf turns into an internal node of the graph.

A VMC includes a set of parameters that influence its dynamics. At the leaves, a set of parameters along with the sensor values determine the rate of Successin production. The amount of Successins can be altered in their way from the leaves to the root, based on a set of parameters

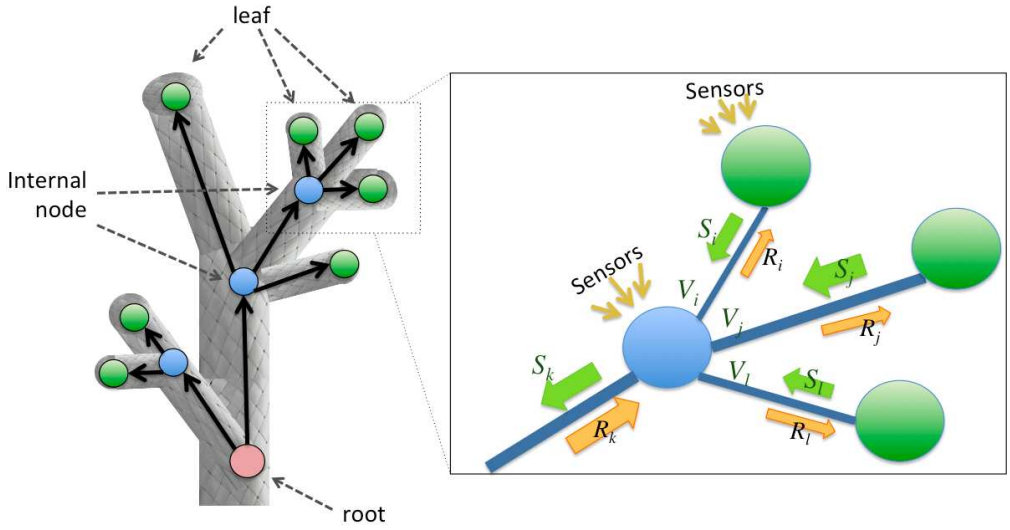


Figure 25: A VMC as a whole is an acyclic directed graph that is distributed on the growing physical structure. The leaves of the graph are the potential places for growth. The leaves produce Successin (S) that flows back towards the root and on its way regulates the thickness of the vessels (V). The vessels are pathways that transport the limited resource (R) from the root to the leaves. The more a leaf gets the resource, the more it is motivated to grow. Sensors and constant parameters of the algorithm contribute to the production and flow of the Ss and regulation of the Vs at every node.

and the sensor values at the internal nodes. The regulation of the vascular pathways at the nodes is based on the value of Successin passing the node as well as a set of parameters. The parameters of the algorithm are identical for all the nodes and can be subject to optimization (e.g., evolutionary algorithms as here) in respect to given objectives for the growing structure.

Every node of VMC runs independently and updates its state variables in parallel to other nodes. The dynamics of the variables, that is, the production of Successin at the leaves and the Successin flow at the internal nodes, the thickness regulation of the vascular pathways, and the distribution of the common limited resource are summarized in Fig. 26.

Successin S_i is produced at a leaf i based on the local sensor values and parameters of the algorithm as follows:

$$S_i := \omega_{\text{const}} + \sum_{s \in \text{sensors}} \omega_s I_s \tag{30}$$

where ω_{const} is a parameter representing the production rate of Successin at the leaf independent of the sensor inputs. I_s is the input from sensor s and ω_s is a parameter determining the weight associated with the input s.

The Successin flows towards the root. The value of S at the internal node i is updated based on the sum of the Successin values arriving from all the children of the node, local sensors, and constant parameters:

$$S_i := g(\rho_{\text{const}} + \sum_{s \in \text{sensors}} \rho_s I_s) \sum_{b \in \text{branches}} S_b, \tag{31}$$

where in the current implementation $g(x) = (1+e^{-10x+5})^{-1}$. The ρ_{const} and ρ_s are transfer rates influencing the rate of reduction in the Successin flow passing a node. ρ_{const} is an independent

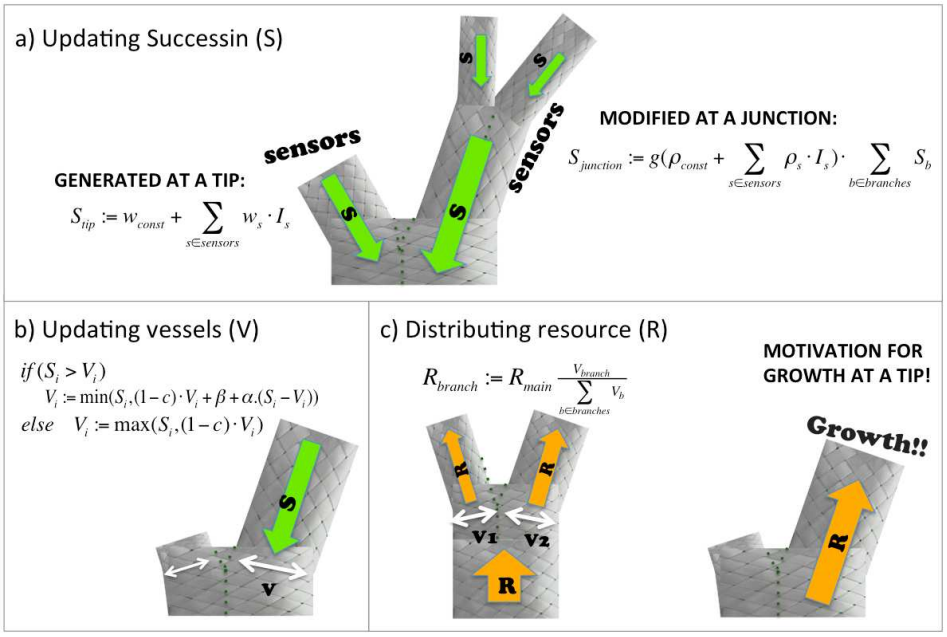


Figure 26: Dynamics of Successin flows, vessel thickness, and resource flows.

rate and ρ_s is the rate associated with a sensor input s. The values of the parameters in this equation contribute to the effect of the distance from the root to the share of the resource reaching the leaves (recall that the values of S passing the nodes adjust the thickness of vessels (V) and consequently influence the distribution of the limited resource between the leaves).

The following equation represents how a vascular pathway V_i is adjusted every time step based on the current value of the Successin passing through it:

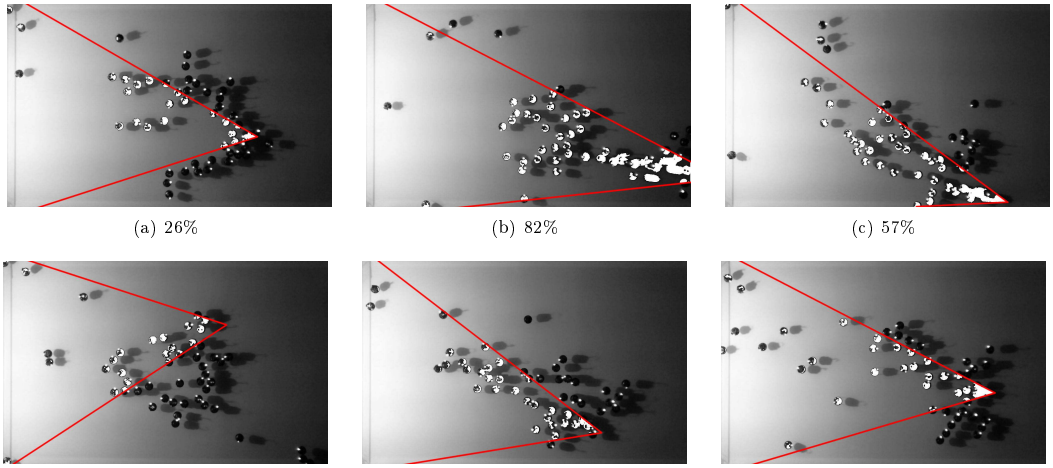
$$V_i := \begin{cases} \min(S_i, (1-c)V_i + \beta + \alpha(S_i - V_i)) & \text{if } S_i \geq V_i \\ \max(S_i, (1-c)V_i) & \text{if } S_i < V_i \end{cases}, \tag{32}$$

where c is a constant decay rate of the vessels, β is the constant addition rate, and α is the factor of adjustment. The equation states that if the current value of S_i is more than the current value of V_i, V_i is likely to increase (depending on the values of the parameters c, β, and α) up to the value of S_i. Otherwise, V_i decreases by a constant decay rate down to the value of S_i. Note that the parameter values in this equation influence the competition between different branches by changing the significance of different values of S in relation to the constant decay and addition rates.

Finally, the limited common resource initiates at the root node and flows towards the leaves. The value is constant at the root. The resource reaching a node m (R_m) is simply divided between its children based on the current value of their vessels:

$$R_i := R_m \frac{V_i}{\sum_{b \in children} V_b}, \tag{33}$$

where R_i is the resource value at the child i of the node m, and *children* is the set of all children of the node (see Fig. 26 for a summary).



(a) 26% (b) 82% (c) 57%

(d) 28% (e) 38% (f) 30%

Figure 27: Kilobots growing towards the light source (left side) using a simple probabilistic approach. Percentages represent the success measurement on the robots' presence within the desired area (red triangle) [11].

Modification: The algorithm is then improved in order to incorporate parameters that better enable the regulation of competition between the branches of the same nodes. In the modified version of the algorithm, the the vascular pathway V_i is updated by using the following equation:

$$V_i := V_i + \alpha(S_i^{f(\text{sensors})} - V_i), \tag{34}$$

$$f(\text{sensors}) := \beta_{\text{const}} \prod_{s \in \text{sensors}} (1 - \beta_s I_s), \tag{35}$$

where α is the adjustment rate of the vessels. β_{const} and β_s are the constant and sensor related parameters that influence competition between difference branches by changing the significance of different values of S for updating the V. The β_{const} provides a base value which is modified by the values of the input sensors and their related β_s. Using the modified version, the competition is easier to regulate either by setting constant parameters or by sensor-dependent dynamics.

In future work, we are planning to use the VMC to control the growth of our braided structures. In a preliminary study, however, we may showcase the algorithm for the assembly and disassembly process among a swarm of Kilobots. The experiment will be on a 2-d surface and light will be used to direct the growth. Earlier, we have shown how a probabilistic adaptive growth process can be implemented on a swarm of 50 Kilobots [11] (see Fig. 27, also see deliverable D2.1).

5.5 Conclusion

A novel distributed morphogenesis controller, called Vascular Morphogenesis Controller (VMC), is proposed here. The controller is inspired from the vessel dynamics and branch competition in plants and is used for driving the morphology of artificial structures. The development of a structure directed by VMC is dynamic and adaptive to environmental changes and environmental constraints. The algorithm is implemented in a growing modular robot in several different setups described in deliverable *D3.1 Representations and design rules.*

6 Conclusion

In this deliverable we have reported various and important results. The VMC approach was proposed for maintaining growth and self-assembly of artificial structures and allow for implementations in distributed systems. The VMC is inspired the growth of plants, which allows us to implement the principles of plant morphology development also in the case of artificial structures in bio-hybrids. A controller of plant growth and motion was presented that allows to steer plants towards user-defined spatial targets. It is more efficiency and more sophisticated compared to the version presented before in D2.1 (M12). Whether this controller is feasible for other species, conditions and scales, will be tested in the future.

The PBDW experiment gives a good overview of the future work of *flora robotica*. The utilization of phototropism for guiding the growth of beans on a diagird structure is an early bio-hybrid setup. As mentioned in Sec. 2, plants affecting procedures are required to be investigated in more detail and subsequently introduced into bio-hybrid. Integration of plants and robots to achieve spatial targets is a great challenge and additional tools combined with advanced controllers for plant shaping will be needed. Future implementations of plant physiology sensors will provide a relevant bridge between living organisms, that are affected by robots, the environment, and human beings. The development of these sensors is in progress.

Investigation on plant signaling in parallel to the development of advanced sensors for signaling, such as electrophysiology sensors, will help to enable more sophisticated plant-robot interactions that go beyond the recent stage of bio-hybrids.

In reference the word 'final' in the title of this deliverable, we can consider the plant shaping controller approach and the VMC approach as final, actually ready-to-use products. However, our ambitious goals in *flora robotica* will require us to further improve them, adapt them to certain application scenarios, and to investigate their performance. Also in the case of procedures for plant treatment, we are able to predict certain effects of plant affection stimulations, but still more experimental work is needed to find applicable dependencies between *flora robotica* robots and possible plant responses.

References

[1] Josh Bongard. Evolutionary robotics. *Commun. ACM*, 56(8):74–83, 2013.

[2] Luzius Brodbeck, Simon Hauser, and Fumiya Iida. Morphological evolution of physical robots through model-free phenotype development. *PLoS ONE*, 10(6):1–17, 06 2015. doi: 10.1371/journal.pone.0128444.

[3] Scott Camazine, Jean-Louis Deneubourg, Nigel R. Franks, James Sneyd, Guy Theraulaz, and Eric Bonabeau. *Self-Organizing Biological Systems*. Princeton University Press, Princeton, NJ, 2001.

[4] Oscar E Checa and Matthew W Blair. Mapping QTL for climbing ability and component traits in common bean (phaseolus vulgaris l.). *Molecular Breeding*, 22(2):201–215, 2008.

[5] Peter Chervenski and Shane Ryan. MultiNEAT, project website;, 2017. URL http://www.multineat.com/. http://www.multineat.com/.

[6] John M. Christie and Angus S. Murphy. Shoot phototropism in higher plants: New light through old concepts. *American Journal of Botany*, 100(1):35–46, 2013. doi: 10.3732/ajb.1200340. URL http://www.amjbot.org/content/100/1/35.abstract.

[7] Kevin R. Cope and Bruce Bugbee. Spectral effects of three types of white light-emitting diodes on plant growth and development: Absolute versus relative amounts of blue light. *HortScience*, 48(4):504–509, 2013. ISSN 00185345.

[8] Sylvain Cussat-Blanc and Jordan Pollack. Cracking the egg: Virtual embryogenesis of real robots. *Artificial Life*, 20(3):361–383, apr 2014. ISSN 1064-5462. URL http://dx.doi.org/10.1162/ARTL_a_00136.

[9] Christelle Der Loughian, Loïc Tadrist, Jean Marc Allain, Julien Diener, Bruno Moulia, and Emmanuel De Langre. Measuring local and global vibration modes in model plants. *Comptes Rendus - Mecanique*, 342(1):1–7, 2014. ISSN 16310721. doi: 10.1016/j.crme.2013.10.010. URL http://dx.doi.org/10.1016/j.crme.2013.10.010.

[10] Mohammad Divband Soorati and Heiko Hamann. The effect of fitness function design on performance in evolutionary robotics: The influence of a priori knowledge. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 153–160. ACM, 2015.

[11] Mohammad Divband Soorati and Heiko Hamann. Robot self-assembly as adaptive growth process: Collective selection of seed position and self-organizing tree-structures. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2016)*, pages 5745–5750. IEEE, 2016. doi: 10.1109/IROS.2016.7759845.

[12] René Doursat, Carlos Sánchez, Razvan Dordea, David Fourquet, and Taras Kowaliw. Embryomorphic engineering: Emergent innovation through evolutionary development. In René Doursat, Hiroki Sayama, and Olivier Michel, editors, *Morphogenetic Engineering: Toward Programmable Complex Systems*, pages 275–311. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-33902-8. doi: 10.1007/978-3-642-33902-8_11.

[13] René Doursat, Hiroki Sayama, and Olivier Michel. A review of morphogenetic engineering. *Natural Computing*, 12(4):517–535, 2013. ISSN 1572-9796. doi: 10.1007/s11047-013-9398-1. URL http://dx.doi.org/10.1007/s11047-013-9398-1.

[14] Peter Eggenberger. Evolving morphologies of simulated 3d organisms based on differential gene expression. In *Proceedings of the fourth european conference on Artificial Life*, pages 205–213. MIT Press, 1997.

[15] Pablo Funes and Jordan B. Pollack. Evolutionary body building: Adaptive physical designs for robots. *Artificial Life*, 4(4):337–357, 1998.

[16] Paco Calvo Garzón and Fred Keijzer. Plants: Adaptive behavior, root-brains, and minimal cognition. *Adaptive Behavior*, 19(3):155–171, 2011.

[17] Brian Goodwin. *How the Leopard Changed Its Spots: The Evolution of Complexity*. Princeton University Press, Princeton, NJ, 2001.

[18] Daniel Nicolas Hofstadler, Mostafa Wahby, Mary Katherine Heinrich, Heiko Hamann, Payam Zahadat, Phil Ayres, and Thomas Schmickl. Evolved control of natural plants: Crossing the reality gap for user-defined steering of growth and motion. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 2017. (submitted).

[19] Sander W. Hogewoning, Emilie Wientjes, Peter Douwstra, Govert Trouwborst, Wim van Ieperen, Roberta Croce, and Jeremy Harbinson. Photosynthetic Quantum Yield Dynamics: From Photosystems to Leaves. *The Plant Cell*, 24(5):1921–1935, 2012. ISSN 1040-4651. doi: 10.1105/tpc.112.097972.

[20] Gregory S. Hornby and Jordan B. Pollack. Body-Brain Co-evolution Using L-systems as a Generative Encoding. In Lee Spector, Erik D. Goodman, Annie Wu, W. B. Langdon, Hans M. Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 868–875, San Francisco, California, USA, Jul-Nov 2001. Morgan Kaufmann.

[21] Annika E. Huber and Taryn L. Bauerle. Long-distance plant signaling pathways in response to multiple stressors: The gap in knowledge. *Journal of Experimental Botany*, 67(7):2063–2079, 2016. ISSN 14602431. doi: 10.1093/jxb/erw099.

[22] Klaus Kalthoff. Pattern formation in early insect embryogenesis - data calling for modification of a recent model. *Journal of Cell Science*, 29(1):1–15, 1978.

[23] Sylvain Koos, Jean-Baptiste Mouret, and Stéphane Doncieux. The transferability approach: Crossing the reality gap in evolutionary robotics. *IEEE Transactions on Evolutionary Computation*, 17(1):122–145, 2013.

[24] Taras Kowaliw and Wolfgang Banzhaf. Mechanisms for complex systems engineering through artificial development. In René Doursat, Hiroki Sayama, and Olivier Michel, editors, *Morphogenetic Engineering: Toward Programmable Complex Systems*, pages 331–351. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-33902-8. doi: 10.1007/978-3-642-33902-8_13. URL http://dx.doi.org/10.1007/978-3-642-33902-8_13.

[25] Antoine Larrieu, Antony Champion, Jonathan Legrand, Julien Lavenus, David Mast, Géraldine Brunoud, Jaesung Oh, Soazig Guyomarc'h, Maxime Pizot, Edward E Farmer, Colin Turnbull, Teva Vernoux, Malcolm J Bennett, and Laurent Laplaze. A fluorescent hormone biosensor reveals the dynamics of jasmonate signalling in plants. *Nature communications*, 6:6043, 2015. ISSN 2041-1723. doi: 10.1038/ncomms7043. URL http://dx.doi.org/10.1038/ncomms7043.

[26] Philip M. Lewis, Lauren N. Ayton, Robyn H. Guymer, Arthur J. Lowery, Peter J. Blamey, Penelope J. Allen, Chi D. Luu, and Jeffrey V. Rosenfeld. Advances in implantable bionic devices for blindness: A review. *ANZ Journal of Surgery*, 86:654–659, 2016. ISSN 14452197. doi: 10.1111/ans.13616.

[27] Aristid Lindenmayer. Developmental algorithms for multicellular organisms: A survey of L-systems. *Journal of Theoretical Biology*, 54(1):3–22, 1975.

[28] Mohsen Mansouryar and Amin Hedayati. Smoothing via iterative averaging (SIA): a basic technique for line smoothing. *International Journal of Computer and Electrical Engineering*, 4(3):307, 2012.

[29] Bernard Millet and Pierre-Marie Badot. The revolving movement mechanism in phaseolus: new approaches to old questions. *Vistas on biorhythmicity*, pages 77–98, 1996.

[30] Paul Moubarak and Pinhas Ben-Tzvi. Modular and reconfigurable mobile robotics. *Robotics and Autonomous Systems*, 60(12):1648–1663, 2012. ISSN 0921-8890. URL http://dx.doi.org/10.1016/j.robot.2012.09.002.

[31] Andrew L. Nelson, Gregory J. Barlow, and Lefteris Doitsidis. Fitness functions in evolutionary robotics: A survey and analysis. *Robotics and Autonomous Systems*, 57:345–370, 2009.

[32] Michitaka Notaguchi and Satoru Okamoto. Dynamics of long-distance signaling via plant vascular tissues. *Frontiers in Plant Science*, 6(161), 2015. ISSN 1664-462X. doi: 10.3389/fpls.2015.00161.

[33] Ronald Pierik and Mieke De Wit. Shade avoidance: Phytochrome signalling and other aboveground neighbour detection cues. *Journal of Experimental Botany*, 65(11):2815–2824, 2014. ISSN 14602431. doi: 10.1093/jxb/ert389.

[34] T. Sachs. How can plants choose the most promising organs? In František Baluška, Stefano Mancuso, and Dieter Volkmann, editors, *Communication in Plants: Neuronal Aspects of Plant Life*, pages 53–63. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-28516-8. doi: 10.1007/978-3-540-28516-8_4.

[35] Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM National Conference*, ACM '68, pages 517–524, New York, NY, USA, 1968. ACM. doi: 10.1145/800186.810616. URL http://doi.acm.org/10.1145/800186.810616.

[36] Karl Sims. Evolving 3D morphology and behavior by competition. In R. Brooks and P. Maes, editors, *Artificial Life IV*, pages 28–39. MIT Press, 1994.

[37] Tomasz Skrzypczak, Rafa Krela, Shraddha Wadurkar, Wojciech Kwiatkowski, Aleksandra Smoczyska, and Przemysawt Wojtaszek. Do plants dream of electronic companion? Plant sciences view on bio-hybrid development. *Frontiers in Robotics and AI*, 2017. (submitted).

[38] Paul L. Snyder, Rachel Greenstadt, and Giuseppe Valetto. Myconet: A fungi-inspired model for superpeer-based peer-to-peer overlay topologies. In *Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems, SASO 2009, San Francisco, California, USA, September 14-18, 2009*, pages 40–50, 2009. doi: 10.1109/SASO.2009.43.

[39] Kenneth O. Stanley and Risto Miikkulainen. Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research*, 21(1):63–100, January 2004.

[40] Maria Stolarz. Circumnutation as a visible plant action and reaction: physiological, cellular and molecular basis for circumnutations. *Plant signaling & behavior*, 4(5):380–387, 2009.

[41] Ichiro Terashima, Takashi Fujita, Takeshi Inoue, Wah Soon Chow, and Riichi Oguchi. Green light drives leaf photosynthesis more efficiently than red light in strong white light: Revisiting the enigmatic question of why leaves are green. *Plant and Cell Physiology*, 50(4): 684–697, 2009. ISSN 00320781. doi: 10.1093/pcp/pcp034.

[42] Anthony Trewavas. *Plant behaviour and intelligence*. Oxford University Press, 2014. ISBN 9780199539543.

[43] Sebastian von Mammen and Christian Jacob. Genetic swarm grammar programming: Ecological breeding like a gardener. In Dipti Srinivasan and Lipo Wang, editors, *2007 IEEE Congress on Evolutionary Computation*, pages 851–858, Singapore, 25-28 September 2007. IEEE Computational Intelligence Society, IEEE Press. ISBN 1-4244-1340-0. doi: doi:10.1109/CEC.2007.4424559.

[44] Mostafa Wahby, Daniel Nicolas Hofstadler, Mary Katherine Heinrich, Payam Zahadat, and Heiko Hamann. An evolutionary robotics approach to the control of plant growth and motion: Modeling plants and crossing the reality gap. In *Self-Adaptive and Self-Organizing Systems (SASO), 2016 IEEE 10th International Conference on*, pages 21–30. IEEE, 2016.

[45] Mostafa Wahby, Daniel Nicolas Hofstadler, Mary Katherine Heinrich, Ewald Neufeld, Igor Kuksin, Payam Zahadat, Thomas Schmickl, Phil Ayres, and Heiko Hamann. Shaping natural plants by distributed robots: Towards bio-hybrids for construction. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2017. (submitted).

[46] Lewis Wolpert. One hundred years of positional information. *Trends in Genetics*, 12(9):359–364, 1996. ISSN 0168-9525. doi: DOI:10.1016/S0168-9525(96)80019-9. URL http://www.sciencedirect.com/science/article/B6TCY-46TB97V-N/2/97cfa0c70a6d2a5b3615d54fa73e81be.

[47] Shu Yuan and Hong-Hui Lin. Role of salicylic acid in plant abiotic stress. *Zeitschrift fur Naturforschung. Teil C: Biochemie, Biophysik, Biologie, Virologie*, 63(5-6):313–320, 2008. ISSN 0939-5075. doi: 10.1515/znc-2008-5-601.

[48] Payam Zahadat and Thomas Schmickl. Generation of diversity in a reaction-diffusion-based controller. *Artificial Life*, 20(3):319–342, 2014.

[49] Payam Zahadat, David Johan Christensen, Serajeddin Katebi, and Kasper Støy. Sensor-coupled fractal gene regulatory networks for locomotion control of a modular snake robot. In *Proceedings of the 10th Int. Symposium on Distributed Autonomous Robotic Systems*, pages 517–530. Springer, 2010. ISBN 978-3-642-32723-0. doi: 10.1007/978-3-642-32723-0_37. URL http://dx.doi.org/10.1007/978-3-642-32723-0_37.

[50] Lingqiu Zeng, Hongchun Qu, Qingsheng Zhu, and Youlan Wang. Incorporating graph automata into plant growth simulation with nutrients transport. *JDCTA*, 4(8):15–22, 2010.