An Evolutionary Robotics Approach to the Control of Plant Growth and Motion: Modeling Plants and Crossing the Reality Gap

Mostafa Wahby*, Daniel Nicolas Hofstadler†, Mary Katherine Heinrich‡, Payam Zahadat†, Heiko Hamann*§
*Heinz Nixdorf Institute, Department of Computer Science, University of Paderborn, Germany
{mostafa.wahby, heiko.hamann}@uni-paderborn.de
†Artificial Life Lab of the Department of Zoology, Karl-Franzens University Graz, Austria
{daniel.hofstadler, payam.zahadat}@uni-graz.at
‡Centre for IT and Architecture, School of Architecture, Royal Danish Academy, Copenhagen, Denmark
mhei@kadk.dk
§Department of Computer Science, Chemnitz University of Technology, Germany

***Abstract*—The self-organizing bio-hybrid collaboration of robots and natural plants allows for a variety of interesting applications. As an example we investigate how robots can be used to control the growth and motion of a natural plant, using LEDs to provide stimuli. We follow an evolutionary robotics approach where task performance is determined by monitoring the plant's reaction. First, we do initial plant experiments with simple, predetermined controllers. Then we use image sampling data as a model of the dynamics of the plant tip xy position. Second, we use this approach to evolve robot controllers in simulation. The task is to make the plant approach three predetermined, distinct points in an xy-plane. Finally, we test the evolved controllers in real plant experiments and find that we cross the reality gap successfully. We shortly describe how we have extended from plant tip to many points on the plant, for a model of the plant stem dynamics. Future work will extend to two-axes image sampling for a 3-d approach.**

1. Introduction

In the context of the EU-funded project *flora robotica* [1], [2], we are interested in creating a self-organizing bio-hybrid [3] that combines the behavior of autonomous robots and living, natural plants. Our long-term objectives are to create mixed societies of growing plants and robotic structures and to generate synergies between them by bringing together the best aspects of both worlds (cf. work on mixed societies of robots and animals [4], [5], [6], [7], [8]). Plants can grow to produce structures efficiently. They sense features of their environment and adapt to dynamic environments [9]. The robots, in turn, can influence the growth of plants by imposing stimuli and can extend the plants' sensing and decision-making capabilities. The challenges of this mixed-society approach are equally distributed between the tasks of sensing and actuation. The plant's state needs to be detected by the robot to allow for closed-loop control. The robot is required to impose appropriate stimuli at appropriate times to influence the plant in the desired way.

Another challenge is the extremely different time scale of plant growth control that differs in several magnitudes from motion control of mobile robots. In comparison to many other living organisms, 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 [10]. In addition to growth, plants also show motion, which is often ignored. Bean shoots' intrinsic motion (circumnutation [11]) allows the plant tips to explore their local environment and – together with phototropism (i.e., directed growth towards or away from light [12]) 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 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 [13].

In this paper, we investigate how the motion and growth of a plant can be directed by a robotic hardware setup that uses light as an attractive stimulus. Before we apply methods of evolutionary robotics we must create an appropriate simulator that addresses relevant features of plant growth. Hence, we first conduct preliminary experiments to generate a simple model for dynamics of a simulated plant tip. Then we use this simple model to evolve robot controllers (i.e., the robot's 'brain' which integrates the sensory input into a coherent reaction via actuator output) that direct the plant motion and growth in agreement to a desired pattern of xy targets.

The variety of plant models in the literature is extensive. Most models from plant science focus on partial aspects of plant systems or are too detailed and too complex for use in robot controllers (e.g., [14]). In the context of research in self-organization and, for example, artificial life, different usefully scaled approaches have been reported to model plant growth. In L-systems [15], a set of rules are iteratively applied to a string of symbols (grammars). The rules process symbols and expand the string whereas certain symbols are interpreted as geometrical structures. Similarly in swarm grammars [16], the L-system is extended such that the reaction of the plants to their environmental stimuli is included in the model. The individual nodes then act as autonomous reactive agents that can be attracted to light sources. Bending of plants (motion) is approximated by considering the stiffness of the connected stem elements and the attraction by the light source in the environment [17]. This way a simple model of plant growth is created that also represents reactions to a dynamic environment. In [18] and [19] abstract branching trees are derived from the inverse computation based on polygon meshes of the geometry of an actual tree and its variations.

Despite this rich variety of available models, we follow a purpose-specific approach that is based on acquired data of actual plant behavior. We find that creating a model specific to our purpose is very efficient and successful as reported in the following. Hence, our approach could serve as a positive example for future applications in similar contexts. In the following we use processed data acquired by sampling images of the growing plant as a simple model of its combined growth and motion behaviors.

In many robotic applications, evolutionary methods were successfully applied to generate controllers for autonomous robots [20]. One approach of evolutionary robotics is that of embodied evolution where the controller is directly evolved on actual robots in hardware [21]. However, the evaluation of an individual's fitness can be a costly and especially time-consuming task. Hence, a simplification is to evolve the controllers in simulations which can implement a considerable speed-up. The drawback is the so-called reality gap problem [22]. It refers to the often experienced problem that controllers developed in simulation may perform poorly in reality due to limitations of the simulation and possibly unknown effects in reality.

In the following, we apply evolutionary methods to evolve a closed-loop controller to direct the tip of a bean plant. The task is to have the plant tip approach predefined points in space by switching a pair of light sources on and off with appropriate order and timing. Controllers are evolved in simulation that we obtain from processed data collected by image sampling in a preliminary experiment setup using real plants. We collect positions of the plant tip along with the current status of the light sources while the light sources are switched on and off in a regular pattern controlled by a trivial, non-reactive controller. From the collected set of plant tip positions we build a simple model to simulate the growth and motion of a bean plant's tip in response to the light sources. A controller for the light sources is then evolved in the simulator for directing the bean's tip such that the tip reaches three different predefined positions. Finally, we use an evolved controller to control a real plant. This test is successful which means we are able to cross the reality gap with our approach.

Figure 1. Bio-hybrid system setup.

We discuss our extension of our image processing method from a single point description of the plant growth tip, to a 10-point description of the full plant stem geometry. This allows the *tip-motion model* to be extended to a full stem-dynamics model. In the future, we will combine this extended method with cameras and image sampling in two axes. This could allow us to evolve robot controllers for more complicated tasks, such as 3-d target patterns or the avoidance of obstacles.

2. Methods

This section describes the setup of the bio-hybrid system, then how it was used to record the positions of bean-tips growing under trivial hand-coded controllers (i.e., light switches in fixed intervals). This data enabled us to construct a model to simulate tip-trajectories in the system under any sequence of light condition changes. With this model in hand, we can evolve controllers on the simulations. We define the task – reaching 3 targets –, present a flexible fitness function and describe the evolutionary approach taken using the MultiNEAT library.

2.1. Bio-hybrid setup

The biological part of the system is the common bean plant (*Phaseolus vulgaris L. var. nanus cf. Saxa*, a bush bean[1]). We germinated the beans in commercial soil for growing vegetables[2] in 1.5l-pots (with 15cm top diameter and soil level at 12cm height).

The robotic part of the system consists of two light sources (Adafruit NeoPixel RGB LED strips[3]) as actuators

1. https://shop.nebelung.de/gemuesesamen/bohnen/buschbohnen-saxa.html
2. FloraSelf Gemüse- und Tomatenerde ohne Torf (Floragard Vertriebs-GmbH)
3. https://www.adafruit.com/products/1506

(plus an additional LED light-bulb[4] as a flash-light), a Raspberry Pi for control, and a camera module[5] as sensor. The Raspberry Pi uses the light sources to control the plant's growth and motion, and receives feedback from the plant through the camera module (see Fig. 1).

The bio-hybrid system is set up in a plant-tent of 200cm height and 120cm in width and depth. The tent is clad in black cloth from the inside to reduce light reflections and to allow for taking high contrast photos. The pot is put on the ground at the center of the back of the tent such that the center of the pot (the location of the root-shoot transition zone) is at a distance of about 8cm to the back wall and 60cm to each side. The camera is set up at a height of 32cm, facing the plant. It is 82cm from the back wall and 74cm from the pot's central axis, with focal plane aligned to that axis. Given a soil level of 12cm, the plant tip is at the center of a captured image when it reaches a height of 20cm.

On each side, we placed a NeoPixel RGB LED strip. A single strip contains 144 individually controllable integrated light sources (a "NeoPixel" or WS2812[6]) each carrying three LEDs: red, green and blue, with peak-emission λ_{max} at wavelengths 630nm, 530nm and 475nm respectively. This means we control 864 LEDs in total, organized in 288 NeoPixels. Each NeoPixel consumes 0.24W when emitting white light at full power, giving 18 lumen. As we always light up one and only one strip, we can expect a total power consumption of up to 8.64A. Each NeoPixel strip is coiled around a cylinder to form an LED strip lamp. The two LED strip lamps are placed on the back-wall of the tent at a height of 30cm from the soil level and with a distance of 35cm to the left and right of the rooting, respectively (see Fig. 1). In addition to the LED strip lamps, a single flash light LED bulb is located directly above the pot at a height of 80cm (68cm above soil level) and is controlled by the Raspberry Pi via a relay. The flash light bulb outputs 806 lumen of warm white light (2700 K) and consumes 9W, however it is only lit up for two seconds once every five minutes.

2.2. Model setup

2.2.1. Preliminary plant experiments. Our plant model is based on experiments with the real plant in a setup with a simplistic, non-reactive controller. An open-loop controller switches between the two light sources in regular time periods of six hours for a total duration of 83.3 hours. We have done six repetitions of this experiment. In each experiment the plant is photographed every five minutes. In Fig. 2 we give compiled images that show the plant's position and geometry during each experiment[7]. They give a good overview of how much motion the plant shows in the horizontal dimension and they also clearly show that the

4. Philips LED bulb 8718696490860 (http://www.philips.co.uk/c-p/8718696490860/)

5. https://www.raspberrypi.org/products/camera-module/

6. WS2812 integrated light source (https://cdn-shop.adafruit.com/datasheets/WS2812.pdf)

7. Find a video at: https://youtu.be/r4PknIwgTyo

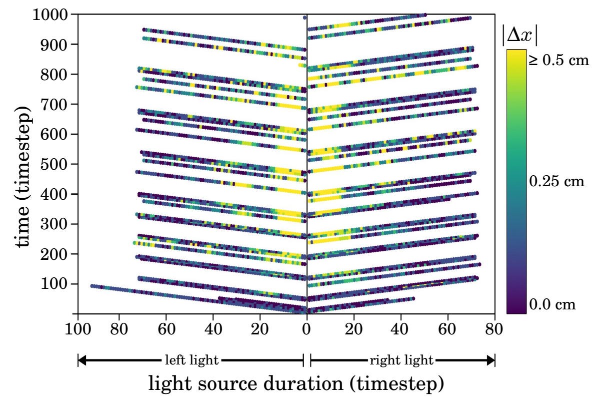


Figure 3. $|\Delta x|$, indicated by color, plotted according to timestep and light source duration, with the right light source accumulating in the positive direction on the x-axis and the left light source accumulating in the negative.

overall plant behavior is equally influenced by growth and motion. In comparing the compiled images of experiments 1, 2, 4, and 6 (experiments with similar durations), a variety in the overall grown height and the horizontal motion is seen, although we have taken care to keep the same experiment setup and conditions. Observing variance in plant experiments is a well-known phenomenon in plant science, which requires high numbers of repetitions. However, in the context of this research, where the focus is on evolutionary computation and robotics, such high overheads for experiments are infeasible. Instead we test our approach based on an engineering perspective by testing whether the model, that results from these experiments, helps us to successfully control a plant. We also test if the evolved controllers are able to perform properly with such dynamic and unexpected plant behavior.

2.2.2. Processing of images. Time-lapse photographs of the above mentioned preliminary experiments are taken and stored at five minute intervals. The images are processed by the following method, using the OpenCV library. First, a Gaussian filter is added to smoothen the images before sampling, thereby reducing error when the sampling resolution is lower than that of the original image. At the sampling resolution, the pixels showing plant material are extracted. The high brightness contrast between background and foreground allows the brightest pixels in the desaturated image to be identified as plant material. Then after cropping the image to the extents of actual plant growth (to exclude the pot and light sources), the plant tip position is extracted. The highest sampled position on the plant is assumed to be the plant tip. The (x, y) plant tip position is stored in cm and scaled to the dimensions of the experiment setup. We make use of this (x, y) position data in our purpose-specific model, described next.

2.2.3. Tip-motion model. We used the data from our preliminary experiments and image processing, to create a simple *tip-motion model*. For simplicity we define the

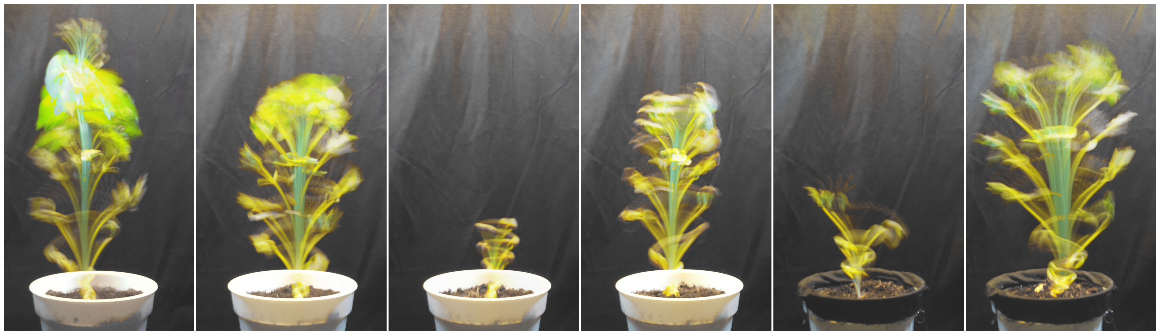
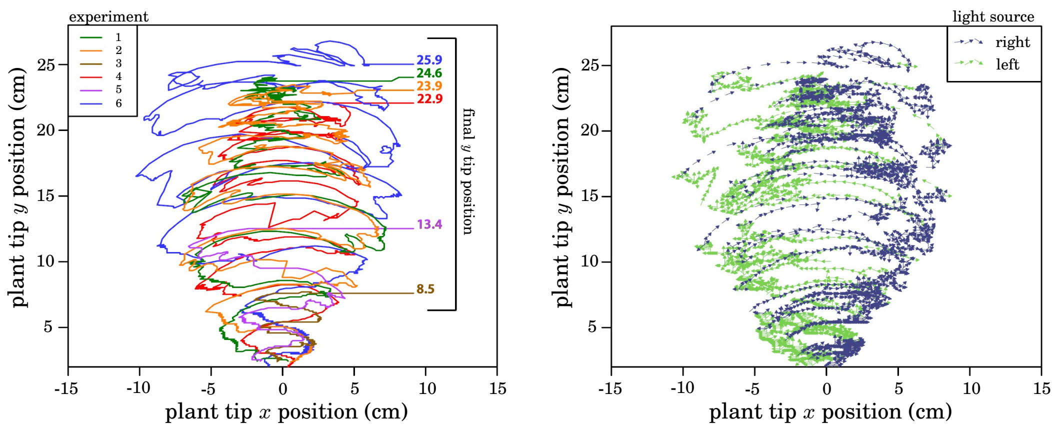


Figure 2. Compiled time-lapse photographs of experiments. From left to right, experiments 1, 2, 3, 4, 5, and 6.



(a) Trajectories of plant tip position in the six preliminary experiments, with experiment number indicated by color (refer to Fig.2) and height in the final timestep in each experiment indicated on the right-hand side.

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

Figure 4. Plant tip trajectories from preliminary experiments (only includes real data; does not include mirrored.)

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 provides the next plant's tip position $\mathbf{x}_{t+1}$, given the current system configuration $(\mathbf{x}, L, C)_t$. Hence, this reality abstraction is used to obtain the next tip position $\mathbf{x}_{t+1}$ for discrete five minutes time steps, until the plant achieves a height of 13.8cm. This simple model captures interesting and relevant behaviors of the plant. For instance, Fig. 3 demonstrates that the plant tip moves slowly in early growth, much more quickly (higher values of $|\Delta x|$) in the middle growth stage (timesteps 300-800), and then more slowly again in late growth.

We assume that the plant has no bias to grow towards either of the two directions (right or left) and also that the two light sources are identical. Therefore, in order to logically double the available data and increase the model's precision, we mirrored the data to both sides by mapping $x \mapsto -x$ and flipping the corresponding Boolean value C while keeping y identical.

Our model calculates the next system configuration $(\mathbf{x}, L, C)_{t+1}$ for a given current system configuration $(\mathbf{x}, L, C)_t$. First, we define a rectangle

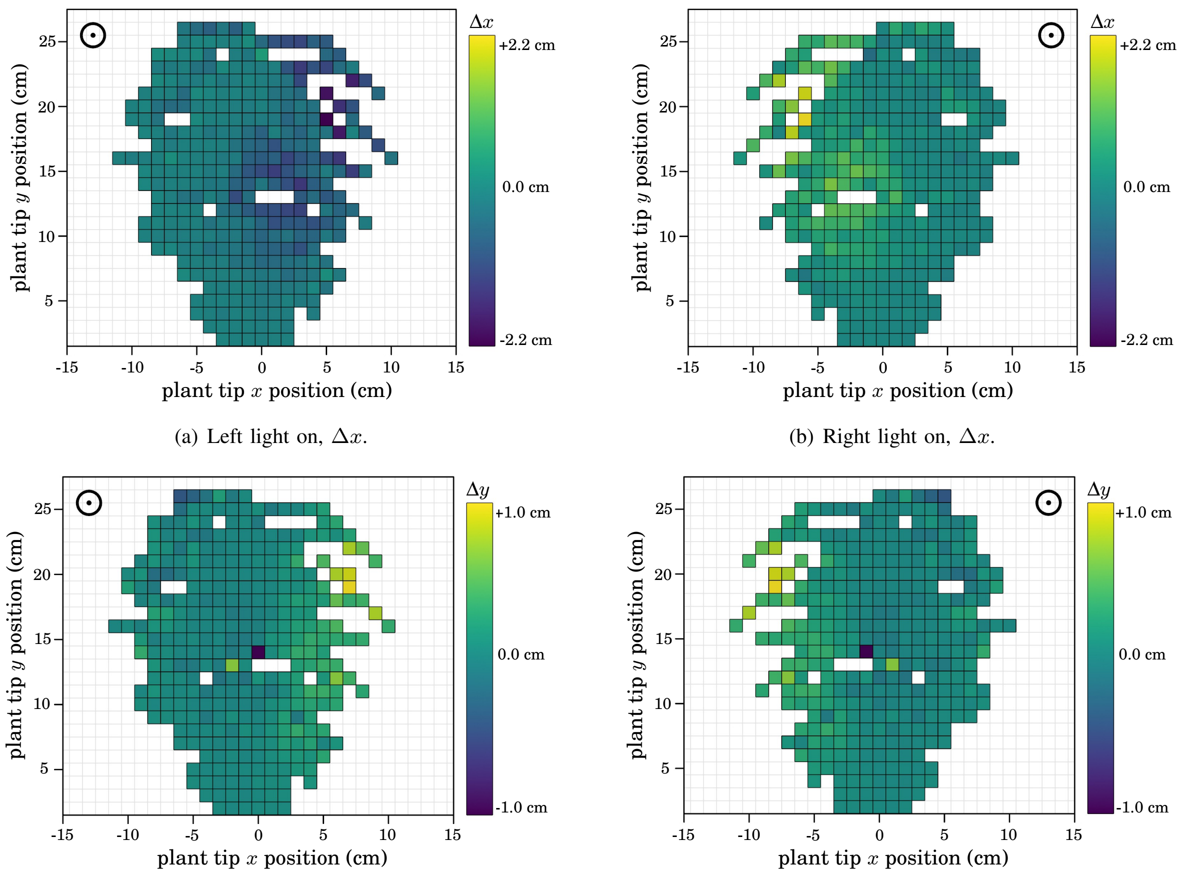
$$R_t = ((x_t - w_x \leq x_t \leq x_t + w_x), (y_t - w_y \leq y_t \leq y_t + w_y)), \quad (1)$$

with the plant tip position $\mathbf{x}_t$ at the center, width $2w_x$, and height $2w_y$ (i.e., a sliding window). In our experiments, we specified a sliding window size of 1cm$\times$2cm ($w_x = 0.5$ and $w_y = 1$). A set P_t of all data points $\hat{\mathbf{x}}$ from our preliminary plant experiments that are within the rectangle R_t and that have the same light condition are selected. From these we collect the set of x-positions only:

$$P_t^x = \{\hat{x} | (\hat{\mathbf{x}} \in R_t) \wedge (\hat{C} = C_t)\}. \quad (2)$$

In addition, we collect the corresponding plant tip positions $\hat{\mathbf{x}}^n$ as they were observed in the subsequent time step in our preliminary plant experiments. We calculate the plant tip shifts $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}^n - \hat{\mathbf{x}}$ for all points in P_t. Our main focus in the model is on the plant tip shifts $\Delta\hat{x} = \hat{x}^n - \hat{x}$ in x-direction. S_t^x is defined as the set of all plant tip shifts in x-direction for P_t^x:

$$S_t^x = \{\Delta\hat{x} | \hat{x} \in P_t^x\}. \quad (3)$$



(a) Left light on, Δx.

(b) Right light on, Δx.

(c) Left light on, Δy.

(d) Right light on, Δy.

Figure 5. Colormaps indicating the distribution of Δx and Δy values at each xy position. Each colormap depicts either Δx data or Δy data, and depicts either timesteps during which the right light was on or timesteps during which the left light was on. All four colormaps include both real data and mirrored data, which are pooled to be used in the model. White patches indicate absence of data.

Fig. 4(a) depicts the xy trajectories of the tip positions from the six preliminary experiments, where data is collected to calculate the plant tip shifts in the model. Fig. 4(b) shows the vector direction of tip motion at each timestep in the experiments, and indicates the portions of each trajectory that occurred when the right light was on or the left light was on (indicated by blue vectors or green vectors, respectively). While these two figures depict the experiment data only, the data used in the model is mirrored, as mentioned before, due to the assumption of having no bias towards left or right (meaning that x data is multiplied by -1 and the light source Boolean value is inverted). Fig. 5 demonstrates the xy distribution of calculated tip shifts (both Δx and Δy) in the model's data (including six real trajectories and six mirrored trajectories). Fig. 5(b) and 5(d) represent the Δx and Δy values when the right light is on. (This includes all right light occurrences in both the mirrored and unmirrored data.) Fig. 5(a) and 5(c) represent the Δx and Δy values when the left light is on (again for both mirrored and unmirrored). These four figures indicate the full set of calculated tip shifts (by light source) for each xy position, which is the data pooled for use in the model.

In our previous work, we have introduced a promising approach which calculates the plant tip shift $\Delta \mathbf{x}_t = (\Delta x_t, \Delta y_t)$ deterministically for the x-coordinate and stochastically for the y-coordinate [23]. The mean $\overline{\Delta \hat{x}} = \frac{1}{|S_t^x|} \sum_{S_t^x} \Delta \hat{x}$ is used to calculate Δx_t while Δy_t is randomly sampled from a normal distribution. This approach has shown promising results, however, it ignores important information about the rotational behavior of the plant tip. It gives changes in y-direction independent of the plant's inclination. As an extension to our former model [23], we now also model plant length L_t and include it to the system configuration. At each time step t, we increment L_t by a randomly sampled value Δy_t from a normal distribution $\mathcal{N}(\mu = 0.03, \sigma = 0.01)$:

$$L_{t+1} = L_t + \Delta y_t. \tag{4}$$

The mean value of 0.03 is the average increase in plant length as observed during all preliminary experiments. Also the variance was estimated based on that data. Hence, for simplicity we assume that the plant grows with the same speed independent of its age. We calculate a temporary plant tip position

$$x'_{t+1} = x_t + \frac{1}{|S_t^x|}\sum_{S_t^x}\Delta\hat{x}, \tag{5}$$

$$y'_{t+1} = y_t + \Delta y_t. \tag{6}$$

Then we determine the resulting inclination angle

$$\alpha_{t+1} = \mathrm{atan2}(y'_{t+1}, x'_{t+1}) \tag{7}$$

of the plant based on this temporary position. In a final step, we determine the plant's actual new tip position

$$\mathbf{x}_{t+1} = \begin{pmatrix} L_{t+1}\cos(\alpha_{t+1}) \\ L_{t+1}\sin(\alpha_{t+1}) \end{pmatrix}. \tag{8}$$

This way we are able to model the rotational movement of the plant tip around its base $(0, 0)$.

2.3. Controller setup

In our setup, the controller is an artificial neural network (ANN). The input to the controller is the current plant tip position $\mathbf{x}_t = (x, y)$ at each time step t (discrete time steps represent five min in reality). The output is a decision, whether the left light or the right light is turned on (the light condition C_t), hence steering the plant tip to achieve the required task. Initially, $\mathbf{x}_0 = (0, 4)$, when the plant is 4cm in height, right after the apical hook has opened and the tip points upward (dicotyledonous plant seedlings typically germinate from the soil with their tip bent downwards for protection of vital tissues [24]). Then, in the case of performing in reality, an image of the plant is captured and processed to acquire the plant tip position. In the case of performing in simulation, the *tip-motion model* is used to acquire the tip position.

2.3.1. Definition of the task. The plant tip controller is required to accomplish a simple task: *three target points*. The plant tip has to approach three different positions in space during the experiments (see Fig. 7). The first target $\mathbf{x}_1^* = (3, 6)$, the second target $\mathbf{x}_2^* = (-5, 9)$ and the third target $\mathbf{x}_3^* = (-1, 13.5)$, while $\mathbf{x}_0^*$ is the initial tip position $\mathbf{x}_0$.

2.3.2. Evolutionary approach. To evolve the controllers, we use MultiNEAT [25]. MultiNEAT is a portable software library implementing NEAT (NeuroEvolution of Augmenting Topologies) in order to evolve ANNs. NEAT [26] is an evolutionary algorithm that attempts to keep the diversity of the population. It starts with simple ANNs and alters both the weights and the network topology by using complexifying methods [26]. Initially, we have evaluated the performance of different sets of NEAT parameters. As a result, the set of parameters in Table 1 has shown better performance (i.e., earlier convergence), hence, was used is our experiments. These parameters were also successful in previous similar experiments [23].

The input layer consists of two neurons, the hidden layer has a variable number of neurons (determined by NEAT), that is, one output neuron, and an unsigned step activation function. The input of the network is the current tip position $(\mathbf{x}_t)$. At each time step t, the experiment's light condition C_t is determined by the network's output. According to the current system configuration $(\mathbf{x}, L, C)_t$ the plant will react to the light stimulus by growth and motion. Next, in the case of performing in simulation, $(\mathbf{x}, L, C)_t$ is passed to the *tip-motion model* to determine the next tip position $(\mathbf{x}_{t+1})$. In the case of performing in reality, an image of the plant is captured and processed and the next tip position is acquired. This procedure is repeated until the plant reaches 13.8cm in height which is enough to allow visiting the three targets and also could be achieved in reasonable period of time for our reality experiments (about 25 hours). The fitness function F (eq. 9) evaluates the performance of the controller.

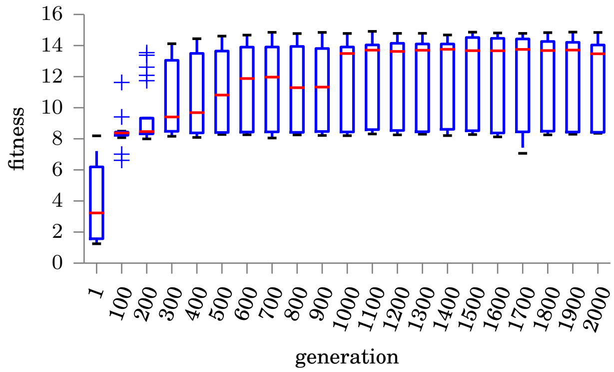
$$\begin{aligned}
\Delta x_t &= x_t - x_{t-1}, \\
f(t) &= \begin{cases} \Delta x_t, & \text{if } x_t < x_i^* \\ -\Delta x_t, & \text{if } x_t > x_i^* \\ |\Delta x_t|, & \text{if } x_t = x_i^* \end{cases}, \\
T_i &= \{t \mid y_{i-1}^* \le y_t < y_i^*\}, \\
F &= \sum_{i=1}^{N}\sum_{t\in T_i} f(t),
\end{aligned} \tag{9}$$

Δx_t is the tip position shift on the x-axis between time steps $t-1$ and t, $f(t)$ is the discrete fitness at time step t, (x_i^*, y_i^*) represent the position of target i, the set T_i is a subset of all the time steps where f is relevant for target i $(y_{i-1}^* \le y_t < y_i^*)$, N is the number of targets (in our experiments $N = 3$), finally, F is the final fitness of the controller after accumulating all the discrete fitnesses for the N targets.

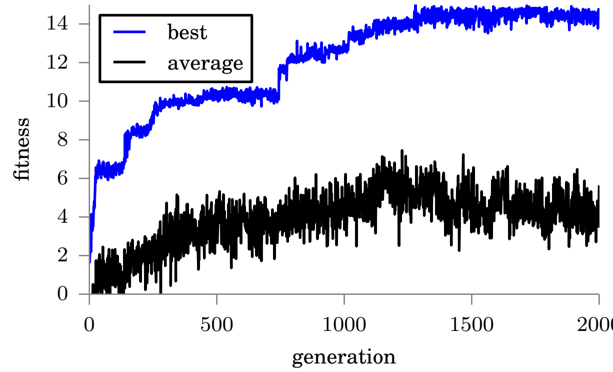
The fitness function is simple and flexible. It is based on a straight forward technique for rewarding/punishing the controller after each time step. If the controller makes the right decision C_t (choose the light source which steers the tip position closer to the next target), it is rewarded a value which is equal to $|\Delta x_t|$ which reflects precisely how well it performed in the current time step. If the controller steers the tip away from the next target, it is given a punishment of $-|\Delta x_t|$. Hence, the theoretical best fitness value is the total distance on the x-axis the plant tip needs to visit all the required targets. For our *three target points* task, the theoretical best is 15cm.

3. Results

First, we report the results of evolving controllers using the above described model (see Sec. 2.2.3). Second, we report the results of experiments that test whether the evolved controllers can be transferred to reality.

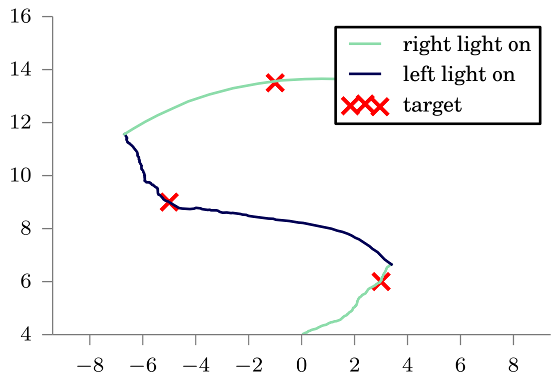


(a) Fitness of best controllers per generation.

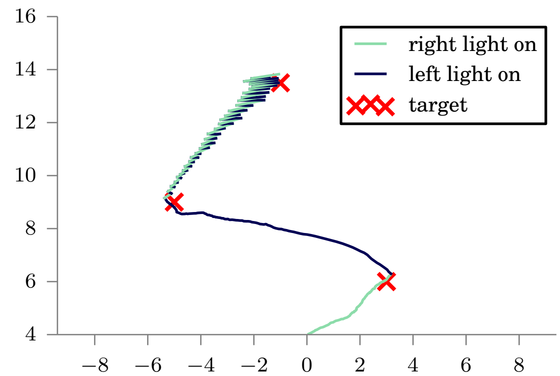


(b) Best and average fitness over generations for a selected evolutionary run.

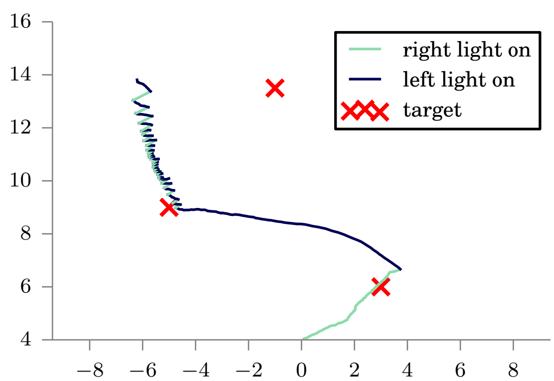
Figure 6. Performance of the evolutionary process over generations for 20 evolutionary runs



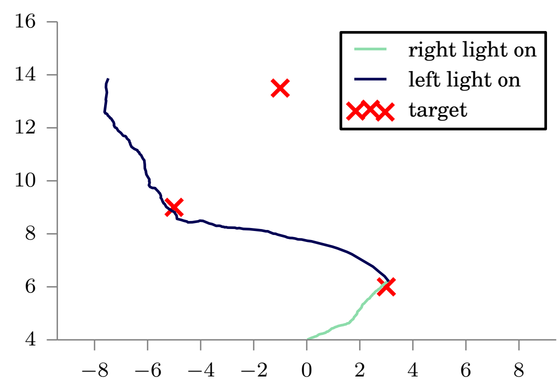
(a) Evolved controller, fitness value = 14.3cm



(b) Evolved controller, fitness value = 14.97cm



(c) Evolved controller, fitness value = 9.13cm



(d) Initial NEAT controller, fitness value = 8.72cm

Figure 7. Trajectories of the simulated plant tip for two successful and two unsuccessful controllers

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

3.1. Evolving controllers in simulation

The controllers are evolved in simulation. The results from 20 independent evolutionary runs are shown in Fig. 6. Fig. 6(a) shows the fitness of the best controller over 2000 generations for all runs (boxplots give minimum, 25th percentile, median, 75th percentile and maximum). Following the median, the fitness increases steadily and saturation is achieved after 1000 generations. Fig. 6(b) shows best and population average fitness for a selected evolutionary run. The first and second observed steps in best fitness correspond to controllers that successfully reach the first and second target points. The third step is a noticeable enhancement in the general performance of the controllers (the tip gets closer to the first two targets and approaches the third one). In the final step, controllers reach the third target followed by a steady enhancement in the general behavior. Fig. 7(a), and 7(b) show two different successful behaviors (i.e., simulated plant tip trajectories) generated by two of the best evolved controllers. The blue color indicates that the left light source is on, while the turquoise color indicates that the right light source is on. In Fig. 7(a), the successful controller keeps the left light on even after reaching the second target point. Only later it switches to the right light source and rotates the plant tip into the third target point mostly by using the motion of the plant. Here, the controller evolved to gradually move away from the third target upon reaching the second target. Later, it switches directions and moves the tip quickly to the third target. This controller scores a fitness of 14.3cm. These seven millimeters are lost because, when the tip reaches the third target's height y_3^*, it is still slightly short of the target's x_3^*. In Fig. 7(b), a different behavior is observed after reaching the second target, such that, the controller keeps switching between the lights until the plant tip reaches the third target. This controller scores a fitness of 14.97cm.

Fig. 7(c) shows the typical trajectory of a controller that fails to reach the third target point. It seems that the model and possibly also the real plant are quite sensitive in the region between target point two and three. At this height and location, according to our model, the plant tip moves faster than usual (higher values of $|\Delta x|$, see Fig. 3). Therefore, when the controller switches directions, the tip moves quickly far beyond the third target, resulting in low fitness despite the achievement of the first two targets.

As mentioned earlier, we have performed 20 evolutionary runs, 2000 generations each. Therefore, we have

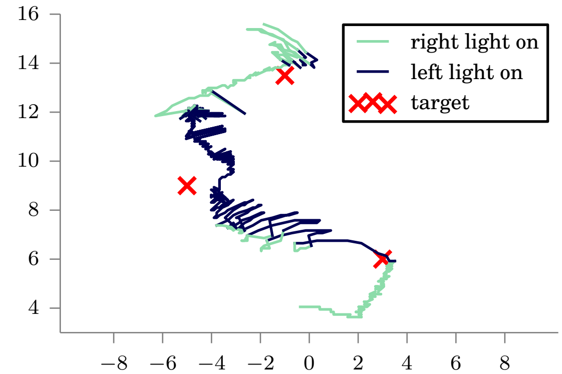


Figure 8. Trajectory of the real plant tip for one of the best evolved controllers, fitness value is 10.9cm.

performed two milion evaluations in total (population size is 50). As a measure of success, we compared our results to the performance of the best controller from two million ANNs obtained from the initialization process of NEAT. The best random controller scores 8.72cm fitness (the tip trajectory is shown in Fig. 7(d)), hence, our evolutionary approach outperforms this random approach.

3.2. Performance of controllers in reality

In a final set of experiments we test whether we can use the controllers that were evolved in simulation also to control a real plant. This is a typical test related to the reality gap problem [22]. Usually, one would expect that a controller evolved in simulation does not directly transfer into a real experiment setup for reasons, such as limitations in the simulation. We tested one of the successful controllers (see Fig. 7(b)) in reality, by running it directly on the Raspberry Pi. The experiment was repeated successfully six times, scoring an average of 9.96cm fitness. The lower fitness is most likely an effect of differences between the plant model and the actual plant behavior. However, the controller seems to compensate and to deal with situations that are different from those seen in the simulations. The tip trajectory of one of the experiments shown in Fig. 8 indicates a reasonable behavior despite the lower fitness. The controller could make the plant tip visit the first and third targets, however, it could only approach the second target. Obviously, the controller was performing the correct behavior by switching on the left light source but the plant tip stopped responding for some while and started growing leaves[8] until it missed the second target. Nevertheless, afterwards, the controller steers the tip to the other direction towards the third target successfully. This controller scored a fitness of 10.9cm in reality (scores 14.97cm in simulation).

Hence, we conclude that the controller successfully transferred from simulation to reality without any changes, that is, we have successfully bridged the reality gap for

8. Find a video at: https://youtu.be/r4PknIwgTyo

this specific problem. We can only speculate about why this approach is successful. The overall task is rather simple and the controller does not need to develop a very sophisticated mapping of plant tip positions to actuator values. However, from the simulation results (see Sec. 3.1) we have learned that especially the region between the second and third target point seems sensitive to the controller's decisions. Our modeling approach is directly based on data obtained in several repetitions of the same setup. Hence, it seems probable that the simulation correctly reflects the reality despite the model's simplicity.

4. Extensions

In ongoing work, we use an extension of the previously described image sampling method for detecting a plant tip throughout growth. Our *tip-motion model*, using a single (x, y) point description of the plant tip, is successful and efficient for the task of reaching a pattern of distinct targets in an xy-plane. As we look to future work of more detailed patterns of targets and more complicated tasks, we seek to extend the single point tip description to a 10-point stem description. These inputs allow us to extend to a model of the full plant stem's dynamics.

The 10-point description of plant stem geometry, as can be seen in Fig. 9, is achieved by a further development of the image sampling method previously described. Time-lapse photographs of the experiments are processed by the following method, using IronPython[9] and the libraries of VPL Grasshopper[10]. These libraries allow the image to be sampled at full resolution and saturation, so no smoothening or desaturation is required. First, the red (R), green (G), blue (B), and brightness (V) channels in the image are individually isolated and remapped onto the domain $(0.0, 1.0)$. Pixels with

$$\begin{aligned}&((G - R \geq 0.5) \wedge (G - B \geq 0.5))\\ &\vee(G \geq 0.75)\\ &\vee(V \geq 0.75)\end{aligned} \tag{10}$$

are identified as plant material and their (w, h) pixel coordinates are saved as (x, y) points. The points are grouped by y-value, and the median x-value is taken for each group. The resulting (x, y) points are identified as plant stem and converted into a planar polyline[11]. Polyline simplification is used to convert the multi-part polyline to a polyline with ten equidistant vertices.

Then timestep history is incorporated, to cull discrete errors caused by variation in lighting conditions of the experiment setups. For this process, the polyline length (L),

9. http://ironpython.net/
10. http://www.rhino3d.com/download/grasshopper/1.0/wip/rc
11. Find a video at: https://youtu.be/r4PknIwgTyo

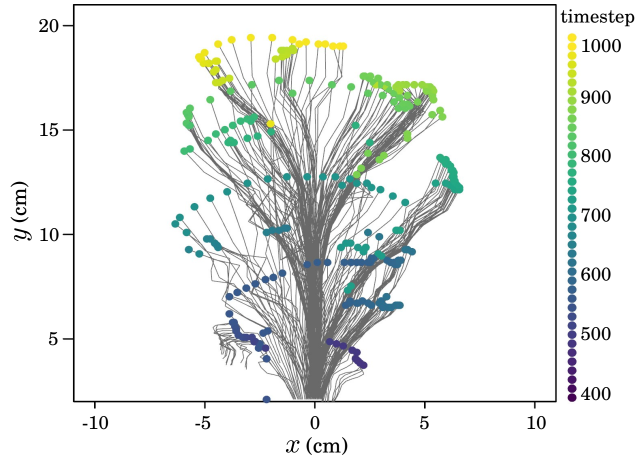


Figure 9. 10 point description, represented as 10-vertex polylines, with end vertex highlighted.

starting vertex (S), and ending vertex (E) are considered. Timestep t is replaced with a duplicate of timestep $t - 1$ if

$$\begin{aligned}&(|L_t - L_{t-1}| \geq 0.25(L_t))\\ &\vee(\text{dist}(S_t, S_{t-1}) \geq 0.25(L_t))\\ &\vee(\text{dist}(E_t, E_{t-1}) \geq 0.5(L_t)).\end{aligned} \tag{11}$$

Afterwards, the 10-vertex polyline is cast as an array of ten (x, y) points, to be used in our extended purpose-specific model of plant stem dynamics. In future work, we will use this extended model for more complicated tasks such as obstacle avoidance or spiralling around an object. The extended model is better suited for these types of tasks because fitness is determined by the behavior of the full stem, not only the tip. By using a 10-point model, information about the plant stem's bending, such as curve radius, apex point of curve, inflection point, or contraflexure point, can be incorporated into the fitness functions used to evole controllers in simulation.

5. Conclusion and future work

We have presented our evolutionary robotics approach to the bio-hybrid collaboration between robots and natural plants. Starting from initial plant experiments, we have made use of a model in our evolutionary runs that is based on image sampling data. Within that model, we have evolved effective controllers that steer the plant's tip to predetermined positions. Hence, we have showed that we can control the growth and motion of a plant with our setup of LEDs as actuators and a camera as sensor. The evolved controllers were tested on real plants and performed correctly which means that our approach successfully bridges the reality gap in this particular setup.

Our future work follows the concepts of the ongoing project *flora robotica* [1], [2]. Besides the obvious next step of using the ten point description of the stem geometry in our next experiments, we also plan to make the last step to a 3-d setup and model. We will need to add another camera, extend the model appropriately, and define a task of either visiting target points in 3-d space or even define tasks, such as growing a spiral shape around an obstacle. Another approach in our future work is to create a decentralized setup with cameras integrated into distributed robotic nodes or relying on proximity sensing only (based on infra-red). In the long run we intend to create *flora robotica* systems with tightly cooperating robots and plants that rely on new levels of self-organized synergetic behavior.

Acknowledgments

Project '*flora robotica*' has received funding from the European Union's Horizon 2020 research and innovation program under the FET grant agreement, no. 640959.

The authors would like to thank Mohammad Divband Soorati for his OpenCV image processing contribution. We also thank Phil Ayres and Thomas Schmickl for their supervisory support.

References

[1] *flora robotica*, "project website," 2016, http://www.florarobotica.eu.

[2] H. Hamann, M. Wahby, T. Schmickl, P. Zahadat, D. Hofstadler, K. Støy, S. Risi, A. Faina, F. Veenstra, S. Kernbach, I. Kuksin, O. Kernbach, P. Ayres, and P. Wojtaszek, "*flora robotica* – mixed societies of symbiotic robot-plant bio-hybrids," in *Proc. of IEEE Symposium on Computational Intelligence (IEEE SSCI 2015)*. IEEE, 2015, pp. 1102–1109.

[3] H. Hamann, Y. Khaluf, J. Botev, M. Divband Soorati, E. Ferrante, O. Kosak, J.-M. Montanier, S. Mostaghim, R. Redpath, J. Timmis *et al.*, "Hybrid societies: challenges and perspectives in the design of collective behavior in self-organizing systems," *Frontiers in Robotics and AI*, 2016.

[4] J. Halloy, G. Sempo, G. Caprari, C. Rivault, M. Asadpour, F. Tâche, I. Saïd, V. Durier, S. Canonge, J. M. Amé, C. Detrain, N. Correll, A. Martinoli, F. Mondada, R. Siegwart, and J. L. Deneubourg, "Social integration of robots into groups of cockroaches to control self-organized choices," *Science*, vol. 318, no. 5853, pp. 1155–1158, Nov. 2007. [Online]. Available: http://dx.doi.org/10.1126/science.1144259

[5] G. Caprari, A. Colot, R. Siegwart, J. Halloy, and J.-L. Deneubourg, "Animal and robot mixed societies: building cooperation between microrobots and cockroaches," *IEEE Robotics & Automation Magazine*, 2005.

[6] R. da Silva Guerra, H. Aonuma, K. Hosoda, and M. Asada, "Behavior change of crickets in a robot-mixed society," *Journal of Robotics and Mechatronics*, vol. 22, no. 4, pp. 526–531, 2010.

[7] A. Gribovskiy, J. Halloy, J.-L. Deneubourg, H. Bleuler, and F. Mondada, "Towards mixed societies of chickens and robots," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, Oct 2010, pp. 4722–4728.

[8] P. Zahadat, M. Bodi, Z. Salem, F. Bonnet, M. E. d. Oliveira, F. Mondada, K. Griparic, T. Haus, S. Bogdan, R. Millsk, P. Marianok, L. Correiak, O. Kernbach, S. Kernbach, and T. Schmickl, "Social adaptation of robots for modulating self-organization in animal societies," in *Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2014 IEEE Eighth International Conference on*, Sept 2014, pp. 55–60.

[9] P. C. Garzón and F. Keijzer, "Plants: Adaptive behavior, root-brains, and minimal cognition," *Adaptive Behavior*, vol. 19, no. 3, pp. 155–171, 2011.

[10] O. E. Checa and M. W. Blair, "Mapping QTL for climbing ability and component traits in common bean (phaseolus vulgaris l.)," *Molecular Breeding*, vol. 22, no. 2, pp. 201–215, 2008.

[11] M. Stolarz, "Circumnutation as a visible plant action and reaction: physiological, cellular and molecular basis for circumnutations," *Plant signaling & behavior*, vol. 4, no. 5, pp. 380–387, 2009.

[12] J. M. Christie and A. S. Murphy, "Shoot phototropism in higher plants: New light through old concepts," *American Journal of Botany*, vol. 100, no. 1, pp. 35–46, 2013. [Online]. Available: http://www.amjbot.org/content/100/1/35.abstract

[13] B. Millet and P. Badot, "The revolving movement mechanism in phaseolus: new approaches to old questions," *Vistas on biorhythmicity*, pp. 77–98, 1996.

[14] R. Bastien, S. Douady, and B. Moulia, "A unified model of shoot tropism in plants: Photo-, gravi- and propio-ception," *PLoS Comput Biol*, vol. 11, no. 2, p. e1004037, 2015.

[15] A. Lindenmayer, "Developmental algorithms for multicellular organisms: A survey of L-systems," *Journal of Theoretical Biology*, vol. 54, no. 1, pp. 3–22, 1975.

[16] S. von Mammen and C. Jacob, "The evolution of swarm grammars – growing trees, crafting art, and bottom-up design," *IEEE Computational Intelligence Magazine*, vol. 4, no. 3, pp. 10–19, 2009.

[17] R. Featherstone and D. Orin, "Robot dynamics: equations and algorithms," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 1, 2000, pp. 826–834 vol.1.

[18] A. Zamuda and J. Brest, "Vectorized procedural models for animated trees reconstruction using differential evolution," *Information Sciences*, vol. 278, pp. 1–21, 2014.

[19] O. Stava, S. Pirk, J. Kratt, B. Chen, R. Měch, O. Deussen, and B. Benes, "Inverse procedural modelling of trees," *Computer Graphics Forum*, vol. 33, no. 6, pp. 118–131, 2014.

[20] J. C. Bongard, "Evolutionary robotics," *Communications of the ACM*, vol. 56, no. 8, pp. 74–83, 2013.

[21] R. A. Watson, S. G. Ficici, and J. B. Pollack, "Embodied evolution: Distributing an evolutionary algorithm in a population of robots," *Robotics and Autonomous Systems*, vol. 39, no. 1, pp. 1–18, 2002.

[22] S. Koos, J.-B. Mouret, and S. Doncieux, "The transferability approach: Crossing the reality gap in evolutionary robotics," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 1, pp. 122–145, 2013.

[23] M. Wahby, M. Divband Soorati, S. von Mammen, and H. Hamann, "Evolution of controllers for robot-plant bio-hybdrids: A simple case study using a model of plant growth and motion," in *Proc. of the 25th Workshop on Computational Intelligence*. KIT Scientific Publishing, 2015, pp. 67–86.

[24] M. Abbas, D. Alabadí, and M. A. Blázquez, "Differential growth at the apical hook: all roads lead to auxin," *Front Plant Sci*, vol. 4, p. 441, 2013. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3817370/

[25] P. Chervenski and S. Ryan, "MultiNEAT, project website;," http://www.multineat.com/. [Online]. Available: http://www.multineat.com/

[26] K. O. Stanley and R. Miikkulainen, "Competitive coevolution through evolutionary complexification," *Journal of Artificial Intelligence Research*, vol. 21, no. 1, pp. 63–100, Jan. 2004.