National Conference on Mechanical Engineering
February 22-23, 2011, Islamic Azad University, Shiraz Branch, Shiraz, Iran

Kinematics and Motion Analysis of a Three-Dimensional Sidewinding Snake-like Robot

S.Sarrafan[1], A. Akbarzadeh[2], S. Molavipoor[3], M. Arhami[4]

[1] Master's Student, Department of Mechanical Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, sarrafan@gmail.com
[2] Associate Professor, Department of Mechanical Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, ali_akbarzadeh@um.ac.ir
[3] Bachelor's Student, Department of Electrical Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, sina1369m@gmail.com
[3] Bachelor's Student, Department of Mechanical Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, mohammad.arhami@yahoo.com

Abstract
In this paper, kinematics of a snake-like robot with maneuverability in three dimensions is calculated using Denavit-Hartenberg and modified Denavit-Hartenberg conventions and is verified by Matlab's Simulink®. A spring-damper system is used to model ground and therefore its normal forces applying to the robot. Coulomb friction is used as the friction model and two different sidewinding gaits are applied to the robot and its motion is compared using three simulation packages, namely Webots®, Simulink® and Adams®.

Keywords: Kinematics – Snake-like robots – Sidewinding – Dynamic Simulation

Introduction
Snake-like robots perform a wide range of movements; they can creep, climb up from obstacles or things like trees, move into pipes or even swim. This different forms of locomotion besides their relatively small cross-section, makes the snake-like robots more useful than other robots for missions in special environments. Moreover since the center of gravity of snake-like robots is near the ground surface, they are fairly stable. On the other hand snake-like robots are hyper-redundant; this property makes them invulnerable to failure of some of their segments. All these characteristics made snake-like robots get employed in missions like search and rescue and inspection, in which sending humans are, by any reason, dangerous or impossible.

Hopkins et al. [1] classified snake-like robots into five groups: robots with passive wheels, robots with active wheels, robots with active treads, robots based on undulation using vertical waves and robots based on undulation using linear expansion. Snake-like robots with passive wheels are the first and most common kind, which were originally presented by inventor of snake-like robots, Dr. Hirose [2]. This robot that was named Active Cord Mechanism 3 (ACM III) had 20 links and to make friction less in the tangential direction, he placed passive wheels under each link. This robot could only move in two dimensions [3]. Later works of Hirose, like ACM-R3 [4] and ACM-R5 [5] are both in this category and are able to move in three dimensions. The ACM-R5 even can swim in a pool. Another example of this kind is AmphiBot I which is built by Crespi et al. [6]. This robot is amphibious and is able to travel both on water and land. This robot is also only able to move in two dimensions. The developed form of the mentioned robot is also built by Crespi and Ijspeert [7] and is called AmphiBot II. The second category of snake-like robots is the ones with active wheels. Koryu-I (KR-I) and Koryu-II (KR-II) are stratified in this category and are consisted of six links [2]. Klsassen and Paap [8] made a six-link robot with the ability to move in three dimensions and named it GMD-Snake2. This robot was the developed version of their previous robot. ACM-R4 which was built by Hirose [3] and The National Technical University of Athens (NTUA) robotic snake [9] are both good examples of the available snake robots in this class. Active treads enable snake-like robots to use treads in addition to the conventional mechanism which the real snake use. By means of these treads, robots are able to traverse extremely rough terrains and debris. OmniTread OT-4 [10] and OmniTread OT-8 [11] which are designed by Borenstein et al. are the best examples of this category of snake-like robots. Robots based on undulation using vertical waves do not need wheels. In this category one can name the ten-link robot introduced by Dowling in 1997 [12] or CMU [13] modular snake robot which both are built in

Carnegie Mellon University or PolyBot reconfigurable robot introduced by Palo Alto Research Center (PARC) [14]. At the end for the examples of robots based on undulation using linear expansion we can name Slim Slime Robot [15] that has six modules and is able to move in three dimensions. It is also worthy to mention snake-like robots which are able to move in three dimensions and are categorized into none of the above classes, like Wheeko [16], KulKo [16], Aiko [17] and the robot introduced by Liljeback et al. [18].

As it is mentioned before, the number of snake-like robots designed to move in three dimensions are far less than snake-like robots with the ability to move in only two dimensions. The most important reason for this fact is the difficult dynamics of these robots. In most cases, the snake-like robot is only operated by the input motor angles and not by motors torque derived by solving dynamics equations. Only a few works have been done on dynamics of a snake-like robot with the ability to move in three dimensions. Ma et al. [19] solved three dimensional dynamics of a snake-like robot using Newton-Euler method, but just for robots that move in two dimensions. Wang et al. [20] modeled dynamics for locomotion-manipulation of a snake-like robot using geometric methods. Transeth et al. also did non-smooth 3D modeling of a snake robot with frictional unilateral constraints and with external obstacles [21,22].

In this paper, kinematics of a snake-like robot with maneuverability in three dimensions is calculated using Denavit-Hartenberg convention and modified Denavit-Hartenberg and is then verified by Matlab's SimMechanics®. A sidewinding gait is applied and the motion of the robot is compared using three simulation packages: Webots®, Simulink® and Adams®. For the Simulink's SimMechanics, a spring-damper system is used to model the normal and frictional forces applying to the robot. These are the first steps of solving dynamic equations of a snake-like robot with maneuverability in three dimensions. It should be also mentioned that the sidewinding is the use of continuous and alternating waves of lateral bending. A downward force is exerted for purchasing on low shear surfaces like sand or loose soil; this mode establishes rolling static contacts to cross relatively smooth substrates.

Kinematics

To solve kinematics of the snake-like robot, we derived Denavit-Hartenberg table and calculated transformation matrices of each consecutive joint and the total transformation matrix. To prove that the solution is correct, we derived these rotation matrices from two methods: Denavit-Hartenberg method and Modified Denavit-Hartenberg method. To get assured we verified our results with SimMechancis toolbox of Matlab.

Our snake-like robot is consisted of 6 links. Each link is connected to the previous one using a universal joint which adds two degrees of freedom. The head of our robot can freely move in the space; hence it has six degrees of freedom. Each universal joint can be modeled as two revolute joints with zero distance from each other and the head of our robot can be modeled as three revolute joints plus three prismatic joints with zero distance from one another.

Denavit-Hartenberg Method

To use this method, we need to put and name our coordinate systems. Here are the conventions we shall follow:

1. Identify the joint axes and imagine (or draw) infinite lines along them. For steps 2 through 5 below, consider two of these neighboring lines (at axes *i* and *i+1*).
2. Identify the common perpendicular between them, or point of intersection. At the point of intersection, or at the point where the common perpendicular meets the i^{th} axis, assign the link-frame origin.
3. Assign the Z_i axis pointing along the i^{th} joint axis.
4. Assign the X_i axis pointing along the common perpendicular, or, if the axes intersect, assign X_1 to be normal to the plane containing the two axes.
5. Assign the Y_i axis to complete a right-hand coordinate system.
6. Assign {0} to match {1} when the first joint variable is zero. For {N}, choose an origin location and X_N direction freely, but generally so as to cause as many linkage parameters as possible to become zero.

Then the Denavit-Hartenberg parameters are computed this way:

a_i = the distance from Z_i to Z_{i+1} measured along X_i;

α_i = the angle from Z_i to Z_{i+1} measured about X_i;

d_i = the distance from X_{i-1} to X_i measured along Z_i;

θ_i = the angle from X_{i-1} to X_i measured about Z_i [23]

Although a modification should be made as d_i should be the distance of origins instead of two consecutive X axes.

In this way this DH table is computed from the coordinate systems depicted in the Figure 1.

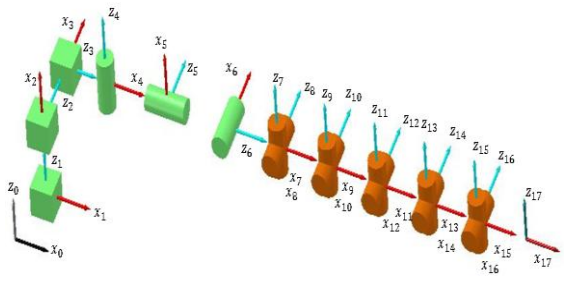


Figure 1. Coordinate systems for Denavit-Hartenberg convention

The resulting Denavit-Hartenberg table is shown in Table 1.

Table 1. Denavit-Hartenberg table

i	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	q_1	0
2	0	$-\pi/2$	q_2	$-\pi/2$
3	0	$-\pi/2$	q_3	$-\pi/2$
4	0	$-\pi/2$	0	$q_4-\pi/2$
5	0	$-\pi/2$	0	$q_5-\pi/2$
6	0	$-\pi/2$	0	$q_6-\pi/2$
7	0	$-\pi/2$	0	$q_7-\pi/2$
8	0	$-\pi/2$	0	q_8
9	l	$\pi/2$	0	q_9
10	0	$-\pi/2$	0	q_{10}
11	l	$\pi/2$	0	q_{11}
12	0	$-\pi/2$	0	q_{12}
13	l	$\pi/2$	0	q_{13}
14	0	$-\pi/2$	0	q_{14}
15	l	$\pi/2$	0	q_{15}
16	0	$-\pi/2$	0	q_{16}
17	l	$\pi/2$	0	0

In which l is length of links and q_1 through q_{16} are general coordinates.

Modified Denavit-Hartenberg Method

To use this method we shall follow these conventions:

1. Choose axis Z_i along the axis of joint *i+1*.
2. Locate the origin O_i at the intersection of axis Z_i with the common normal to axes Z_{i-1} and Z_i. Also, locate $O_{i'}$ at the intersection of the common normal with axis Z_{i-1}.
3. Choose axis xi along the common normal to axes Z_{i-1} and Z_i with direction from joint i to joint *i+1*.
4. Choose axis Y_i so as to complete a right-handed frame.
5. For frame 0, only the direction of axis Z_0 is specified; then O_0 and X_0 can be arbitrarily chosen.
6. For frame n, since there is no joint *n+1*, Z_n is not uniquely defined while X_n has to be normal to axis Z_{n-1}. Typically, Joint n is revolute, and thus Z_n is to be aligned with the direction of Z_{n-1}.
7. When two consecutive axes are parallel, the common normal between them is not uniquely defined.
8. When two consecutive axes intersect, the direction of X_i is arbitrary.
9. When joint *i* is prismatic, the direction of Z_{i-1} is arbitrary.

Then we calculate DH parameters this way:

a_i = distance between O_i and $O_{i'}$,
d_i = coordinate of $O_{i'}$ along Z_{i-1},
α_i = angle between axes Z_{i-1} and Z_i about axis X_i to be taken positive when rotation is made counter-clockwise,
θ_i = angle between axes X_{i-1} and X_i about axis Z_{i-1} to be taken positive when rotation is made counter-clockwise.[24]

So by naming coordinate systems like Figure 2, we obtained modified DH parameters which are shown in Table 2.

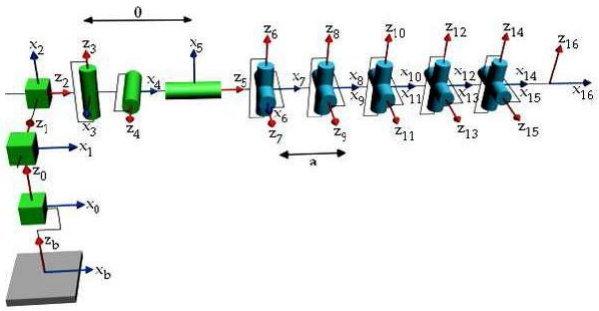


Figure 2. Coordinate systems for modified Denavit-Hartenberg convention [18]

Table 2. modified Denavit-Hartenberg parameters

i	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	$-\pi/2$	q_1	0
2	0	$-\pi/2$	q_2	$-\pi/2$
3	0	$\pi/2$	q_3	$\pi/2$
4	0	$\pi/2$	0	$q_4+\pi/2$
5	0	$\pi/2$	0	$q_5+\pi/2$
6	0	$\pi/2$	0	$q_6+\pi/2$
7	0	$\pi/2$	0	$q_7+\pi/2$
8	l	$-\pi/2$	0	q_8
9	0	$\pi/2$	0	q_9
10	l	$-\pi/2$	0	q_{10}
11	0	$\pi/2$	0	q_{11}
12	l	$-\pi/2$	0	q_{12}
13	0	$\pi/2$	0	q_{13}
14	l	$-\pi/2$	0	q_{14}
15	0	$\pi/2$	0	q_{15}
16	l	$-\pi/2$	0	q_{16}

SimMechanics

To verify our results with a rather non-mathematical tool, SimMechanics toolbox has been used. Six bodies are connected to each other via five universal joints. The first joint, which can be interpreted as the robot's head, has the all six possible degrees of freedom. Each body is connected to a sensor and a display box which shows position and rotation matrix of corresponding link. The first joint that is somehow virtual and a free end in reality is connected to ground and machine environment. A figure illustrating the structure of the robot is available in the Appendix I.

Links are designed such that x-axis is along links, z-axis is perpendicular to them and y-axis is vertical to the normal plane, oriented toward inside of it. Because of our joints order in Denavit-Hartenberg kinematics solution, an Euler rotation should be used for each universal joint in which rotation over z-axis happens prior to y-axis.

Sidewinding Gait

As mentioned before, sidewinding is the use of continuous and alternating waves of lateral bending. A downward force is exerted for purchase on low shear surfaces like sand or loose soil; this mode

establishes rolling static contacts to cross relatively smooth substrates. There are only two contact patches while the snake is in motion. The technique minimizes slippage and is even more efficient than lateral undulation. Some sidewinding snakes have been observed to travel kilometer-length distances continuously. Sidewinding is used primarily by snakes in desert regions where loose soils and sands are prevalent. The development of sidewinding may be related both to the need for traction on low shear surfaces such as sand and the need to avoid the high temperatures of desert terrain. As shown in Figure 3, this gait can be thought of as the 'peeling' of the body from one track to the next. The tracks, or lines, show the rolling of the body contacts during locomotion. [25]

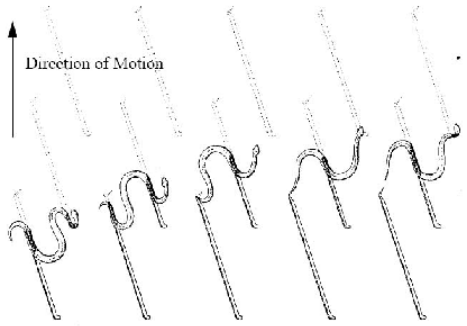


Figure 3. Sidewinding Gait

The twisting angles applied to each link in horizontal and vertical directions by motors in a snake-like robot usually take the form of Equations 1 and 2.

$$\theta_h = A_{hor}\sin(\omega_{hor}t + i\delta_{hor}) + \psi_{hor} \quad (1)$$

$$\theta_{ver} = A_{ver}\sin(\omega_{ver}t + i\delta_{ver} + \delta_0) + \psi_{hor} \quad (2)$$

In which i is the number of joint and other values are set as the Equation 3 for a side winding gait. [18]

$$\begin{aligned} A_{hor} &= 30°, \omega_{hor} = \frac{3\pi}{4}\ rad/s \\ \delta_{hor} &= -70°, \psi_{hor} = 0° \\ A_{ver} &= 30°, \omega_{ver} = \frac{3\pi}{4}\ rad/s \\ \delta_{ver} &= -70°, \psi_{ver} = 0° \end{aligned} \quad (3)$$

δ_0 indicates direction of the movement and is set either 0 or π/2.

Simulation

We have used three different simulation packages to simulate motion of a same snake-like subjected to the mentioned gait. The modeled snake-like robot is consisted of six cylindrical links, each with 20cm length, 300gr mass and 15cm radius. That makes the total length of our robot 1.2m and its total mass 1.8kg approximately. Joints connecting segments to each others are universal joints, providing our robot with an additional two degrees of freedom.

The friction between robot and ground is estimated as coulomb friction between steel-steel which has about 0.8 static friction coefficient.

Webots

Webots is a well-known simulation environment developed by Cyberbotics Ltd., which has been recently used in many robotic researches in order to model and simulate mobile robots [26]. It provides us with a variety of tools, textures, objects and shapes and robots can be programmed by many programming languages such as C, Java and MATLAB.

Each object in Webots has physical parameters such as mass, inertia matrix, density and coulomb friction. Some of most significant so called as "objects" we used in our work are:

- Servo motor: The motors, which for it we can set maximum velocity, maximum force, acceleration and so forth.
- GPS: This module gives us the exact global position of the robot. We send the information of GPS by an emitter and a supervisor receives all information that gives us the ability to analyze the robot motion.
- Emitter: This is a radio emitter by which we can send information through different channels.
- Supervisor: To analyze the whole project as an observer and control robot, this module is used. It has an exact controller same as the robot controller. We have put a receiver for our supervisor to get positions data. [27]

We have designed a 6-link robot with universal joints. In this case, for each link, two servo motor have to be designated which rotates in perpendicular axes. In the final results, location of each link is used. The design procedure is as follow:

I) Design of a single link
II) Connecting links to each others
III) Programming and controlling

The shape is in the form of a cylinder. GPS is in the center of mass of each segment for location measurements. Connectors are objects used for connecting robot parts together; therefore in both ends of links we set a connector which links the previous servo motor to the next body part.

In the Webots environment, for each robot, "controller" is a program (in our case C language) to control the module. The controller codes can be found in the Appendix II. Figure 4 shows the software's user interface.

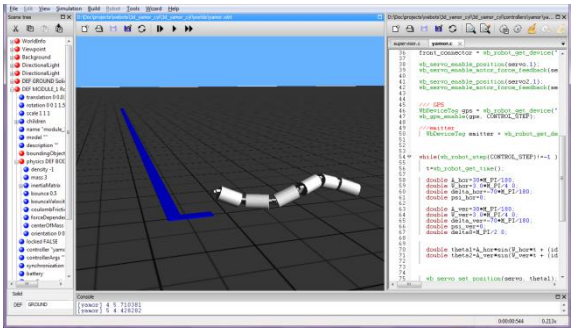


Figure 4. Interface of Webots

Adams

Adams simulation software and its add-on tool, Human Figure Modeler, developed and marketed by Mechanical Dynamics Inc., was, for the first time,

introduced to bioengineering students at the University of Pittsburgh to enhance the learning of biomechanical principles. MSC-Adams is maybe the most powerful software in simulation and analysis of dynamic systems. It helps engineers by its abilities to model, examine, evaluate and optimize dynamic systems before physically manufacturing it. By the help of this software, one can design dynamic system and after the simulation, find its outputs such as displacement, velocity, acceleration, force and torque. Analysis of vibration and modeling flexible structures are other unique capabilities of this package. Besides, it can be coupled to some other technical softwares like Ansys, Catia, Pro/E.
Some of its major modules are:

- Adams/View: For modeling in three dimensions, defining constraints and joints, 3D animation and illustrating forces, displacements, stress and so on.
- Adams/Solver: Numerical solver of modeled dynamical systems in the software which uses Euler-Lagrange method.
- Adams/PostProcessor: This module demonstrates results of other modules. [28][29]

Figure 5 depicts the Adams environment.

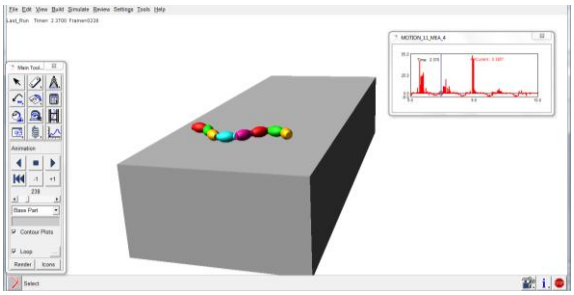


Figure 5. MSC Adams environment

SimMechanics

SimMechanics software is a block diagram modeling environment for the engineering design and simulation of rigid body machines and their motions, using the standard Newtonian dynamics of forces and torques. Mechanical systems are represented by a connected block diagram, like other Simulink models, and hierarchical subsystems can be incorporated. The visualization tools of SimMechanics software display and animate simplified renderings of 3-D machines, before and during simulation, using the MATLAB Graphics system. [30]
Since in SimMechanics there is no ground, we have modeled it using a spring-damper system. As it is illustrated in Figure 6, this mechanism applies to gravity center of links that their z-positions are less than zero.

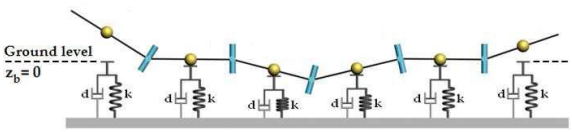


Figure 6.Spring-Damper system used to model the ground [18]

Therefore the normal force formulation will be written as the Equation 4.

$$F_{N,i} = \begin{cases} 0 & \forall p_{CG_{i,z}} \geq 0 \\ -k.p_{CG_{i,z}} - d.v_{CG_{i,z}} & \forall p_{CG_{i,z}} < 0 \end{cases} \quad (4)$$

Where $p_{CG_{i,z}}$ and $v_{CG_{i,z}}$ are the center of gravity of link *i* position and velocity in the *z* direction. *k* and *d* are spring and damper factors respectively.
For friction, coulomb friction model is implemented. As in most snakes, the tangential friction is less than normal friction; two different directions are used for our model which is described in Equation 5.

$$F_{f,i}^{\gamma} = \begin{cases} -\frac{v_i^{\gamma}}{V_{cr}^{\gamma}} \mu_{\gamma s} F_{N,i} & |v_i^{\gamma}| < V_{cr}^{\gamma} \\ -sign(v_i^{\gamma}) \mu_{\gamma D} F_{N,i} & |v_i^{\gamma}| \geq V_{cr}^{\gamma} \end{cases} \quad (5)$$

Where $\gamma = t, n$ which shows tangent and normal directions, v_i^{γ} is the velocity of link i in γ direction. $\mu_{\gamma s}$ and $\mu_{\gamma D}$ are static and dynamic coulomb friction coefficients in γ direction respectively, V_{cr}^{γ} is the critical speed in γ direction and *sign( )* is the signum function.

Results and Discussion

For the three software packages described, the simulation performed for one minute. First we set $\delta_0 = 0$. Figure 7, 8 and 9 shows displacement of the robot after 1 minute in Webots, Adams and SimMechanics.

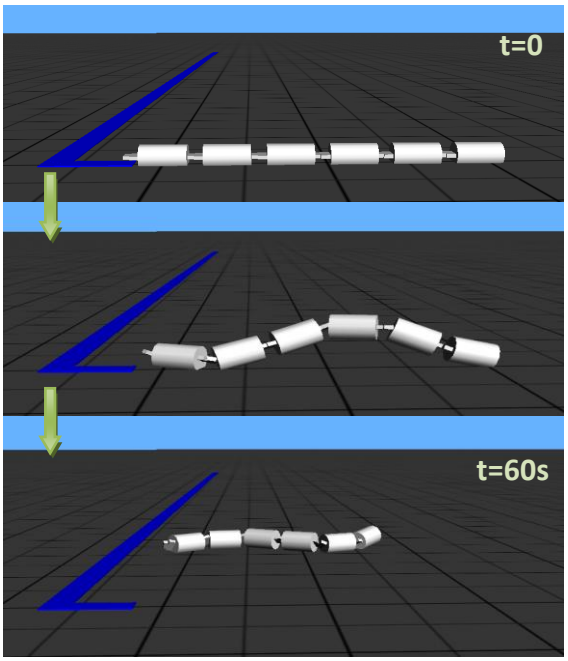


Figure 7. Simulation of the snake-like robot in Webots with $\delta_0 = 0$

The displacements in 60seconds for all three simulations were approximately the same and 1m. The experiment repeated for $\delta_0 = \pi/2$ and the consequent displacement was again approximately the same. Figures 10 and 11 illustrate the movement in Webots and Adams software packages. Since SimMechanics animation is not very graphically understandable, pictures of its simulation are omitted here.

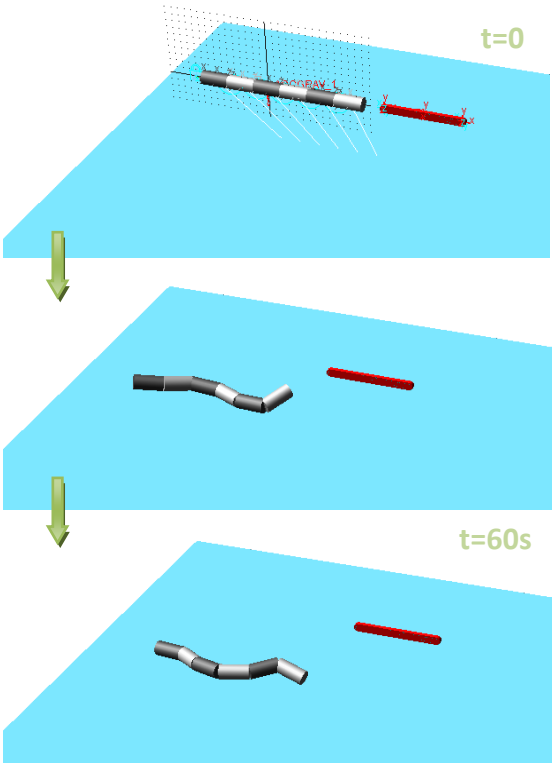


Figure 8. Simulation of the snake-like robot in Adams with $\delta_0 = 0$

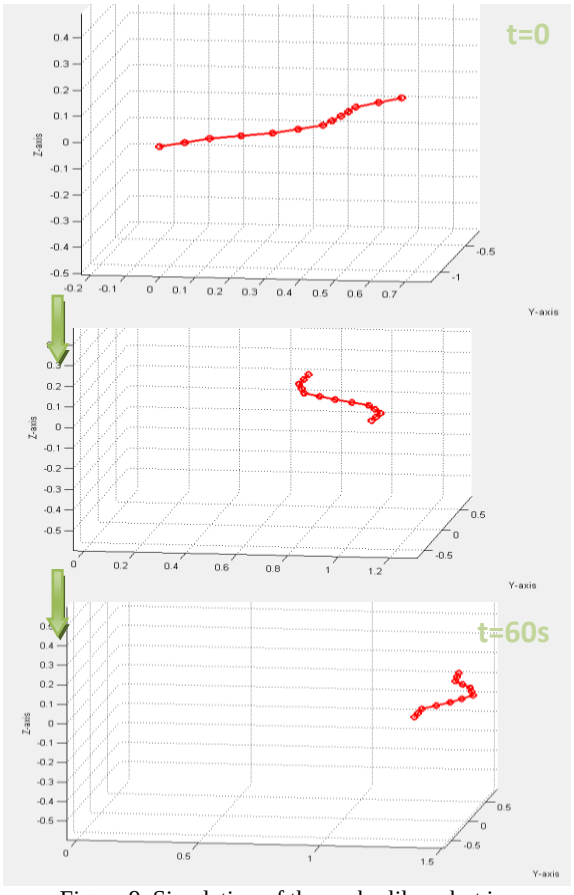


Figure 9. Simulation of the snake-like robot in SimMechancis with $\delta_0 = 0$

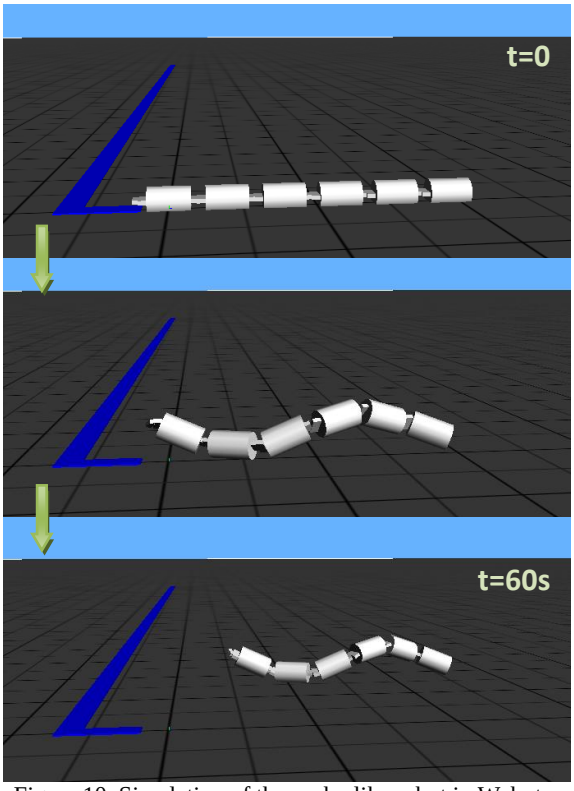


Figure 10. Simulation of the snake-like robot in Webots with $\delta_0 = \pi/2$

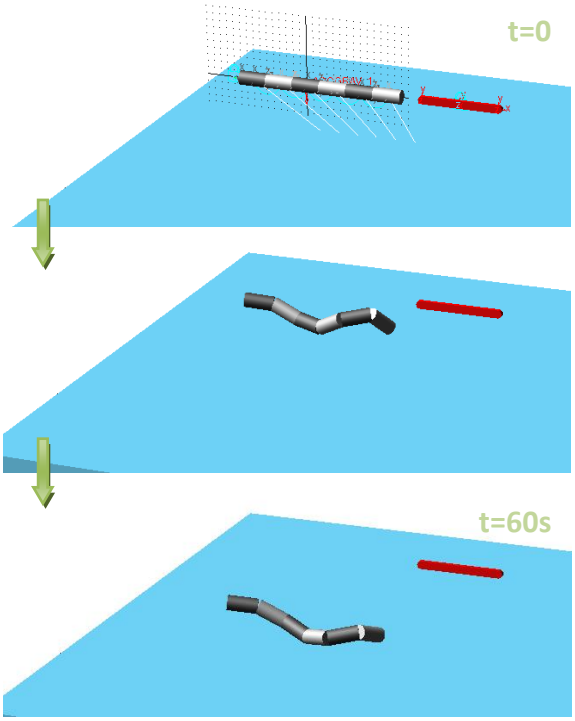


Figure 11. Simulation of the snake-like robot in Adams with $\delta_0 = \pi/2$

References

1- Hopkins, J.K., Spranklin, B.W., and Gupta, S.K.,"A survey of snake-inspired robot designs", 2009, *Bionispiration and Biomimetics*, vol.4.

2- Hirose, S.,"Biologically Inspired Robots: Snake-Like Locomotors and Manipulators", 1993, Oxford University Press.

3- Hirose, S., http://www-robot.mes.titech.ac.jp/robot/snake_e.html

4- Mori, M., and Hirose, S., "Development of Active Cord Mechanism ACM-R3 with Agile 3D Mobility", 2001, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Maui, Hawaii.

5- HiBot. http://www.hibot.co.jp/html/html_eng/products/available.html#ACM-R5

6- Crespi, A.,Badertscher, A., Guignard, A., and Ijspeert, A.J.,"AmphiBot I: an Amphibious Snake-like Robot", 2005, *Robotics and Autonomous Systems*, vol. 50.

7- Crespi, A., and Ijspeert, A.J., "AmphiBot II: An Amphibious Snake Robot that Crawls and Swims using a Central Pattern Generator", 2006, *Proc 9th International Conference on Climbing and Walking Robots*, Brussels, Belgium.

8- Klaassen,B., and Paap, K.L., "GMD-SNAKE2: A Snake-Like Robot Driven by Wheels and a Method for Motion Control", 1999, *IEEE International Conference on Robotics and Automation*, Detroit, Michigan.

9- Kyriakopoulos, K.J., Migadis, G., and Sarrigeorgides, K., "The NTUA Snake: Design, Planar Kinematics, and Motion Planning", 1999, *Journal of Robotic Systems*, vol. 16.

10- Borenstein, J., Hansen, M.G., and Nguyen, H., "The OmniTread OT-4 Serpentine Robot for Emergencies and Hazardous Environments", 2006, *International Joint Topical Meeting: Sharing Solutions for Emergencies and Hazardous Environments*, Salt Lake City, Utah, USA.

11-Borenstein, J., Granosik, G., and Hansen, M., "The OmniTread Serpentine Robot – Design and Field Performance", 2005, *Proc. SPIE Defense and Security Conference, Unmanned Ground Vehicle Technology VII*, Orlando, FL.

12-Dowling, K., "Limbless Locomotion: Learning to Crawl with a Snake Robot", 1997, Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

13-Wright, C., Johnson, A., Peck, A., McCord, Z., Naaktgeboren, A., Gianfortoni, P., Gonzalez-Rivero, M., Hatton, R., and Choset, H., "Design of a Modular Snake Robot", 2007, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA.

14-Yim, M., Duff, D.G., and Roufas, K., "Modular Reconfigurable Robots: An Approach to Urban Search and Rescue", 2000, *Proc. 1st International Workshop on Human-friendly Welfare Robotics Systems*, Taejon, Korea.

15-Ohno, H. and Hirose, S., "Study on Slime Robot (Proposal of Slime Robot and Design of Slim Slime Robot)", 2000, *Proc. the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3.

16-Liljebäck, P., "Modelling, Development, and Control of Snake Robots", 2011, Ph.D. Thesis, Norwegian University of Science and Technology, Trondheim, Norwegen.

17-Transeth, A.A., "Modelling and Control of Snake Robots", 2007, Ph.D. Thesis, Norwegian University of Science and Technology, Trondheim, Norwegen.

18-Liljeback, P., Stavdahl, Ø.,Pettersen, K.Y., "Modular Pneumatic Snake Robot: 3D Modelling, Implementation And Control", 2008, *Modeling, Identification and Control*, vol. 29, No. 1, pp. 21–28.

19-Ma, Sh., Ohmameuda, Y., Inoue, K., "Dynamic Analysis of 3-dimensional Snake Robots", 2004, *Proceedings of 2004 IEEWSl International Conference on Intelligent Robots and Systems*.

20-Wang, Zh., Ma, Sh., Li,B., and Wang, Y., "Dynamic Modeling for Locomotion-manipulation of a Snake-like Robot by Using Geometric Methods", 2009, *The 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

21-Transeth, A.A., Leine, R.,Glocker, Ch., and Pettersen, K. ,"Non-smooth 3D Modeling of a Snake Robot with External Obstacles", 2006, *Proceedings of the 2006 IEEE International Conference on Robotics and Biomimetics*.

22-Transeth, A.A., Leinet, R. ,Glockert, C., and Pettersen, K.Y. ,"Non-smooth 3D Modeling of a Snake Robot with Frictional Unilateral Constraints", 2006, *proceedings of the 2006 IEEE International Conference on Robotics and Biomimetics*, Kunming, China.

23-Craig, J.J., "Introduction to Robotics: Mechanism and Control", 3rd Ed, Pearson Education Inc., 2005.

24-Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G., "Robotics: Modeling, Planning and Control", Springer, 2009.

25-Shmakov, O., "Snakelike robots locomotions control", 2006, Course 5: Mechatronics - Foundations and Applications

26-Cyberbotics. Webots reference manual 6.2.4, 2010.

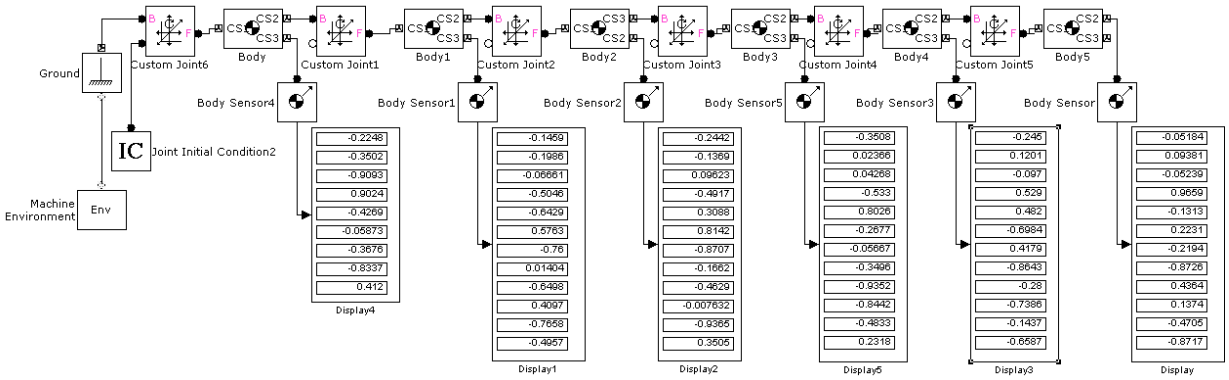
27-Cyberbotics. http://www.cyberbotics.com, Jan 2012.

28- Cham, R., Moyer, B., "Introducing ADAMS, a Mechanical System Simulation Software, to Bioengineering Students", 2002, University of Pittsburgh

29-www.MSCSoftware.com

30-MATLAB's Simulink 7.1 Documentations

Appendix I



Appendix II

```
#include <webots/robot.h>
#include <webots/servo.h>
#include <webots/connector.h>
#include <stdlib.h>
#include <math.h>
#include <stdio.h>
#include <webots/gps.h>
#include <webots/emitter.h>
#define CONTROL_STEP 32
#define N 8
static double t = 0.0;                  /* time elapsed since simulation start [s] */
staticint id = -1;                     /* this module's ID */
staticWbDeviceTag servo,servo2, rear_connector, front_connector;
staticconst double F = 1.0;             /* frequency */
staticconst double A = 0.6;             /* amplitude */
typedefenum { GPS, SUPERVISED } gps_mode_types;
int main() {
wb_robot_init();
const char *name = wb_robot_get_name();
id = atoi(name + 7) - 1;
servo            = wb_robot_get_device("servo");
servo2           = wb_robot_get_device("servo2");
rear_connector  =wb_robot_get_device("rear_connector");
front_connector = wb_robot_get_device("front_connector");
wb_servo_enable_position(servo,1);
wb_servo_enable_motor_force_feedback(servo,1);
wb_servo_enable_position(servo2,1);
wb_servo_enable_motor_force_feedback(servo2,1);
/// GPS
WbDeviceTaggps = wb_robot_get_device("GPS_1");
wb_gps_enable(gps, CONTROL_STEP);
///emitter
WbDeviceTag emitter = wb_robot_get_device("emitter");
while(wb_robot_step(CONTROL_STEP)!=-1 ) {
t=wb_robot_get_time();
doubleA_hor=30*M_PI/180;
doubleW_hor=3.0*M_PI/4.0;
doubledelta_hor=-70*M_PI/180;
doublepsi_hor=0;
doubleA_ver=30*M_PI/180;
doubleW_ver=3.0*M_PI/4.0;
doubledelta_ver=-70*M_PI/180;
doublepsi_ver=0;
double delta0=M_PI/2.0;
double theta1=A_hor*sin(W_hor*t + (id)*delta_hor ) + psi_hor;
double theta2=A_ver*sin(W_ver*t + (id)*delta_ver + delta0 ) + psi_ver;
wb_servo_set_position(servo, theta1);
wb_servo_set_position(servo2, theta2);
///GPS
double *gps_values = wb_gps_get_values(gps);
wb_emitter_send(emitter, gps_values, 3*sizeof(double));
}
wb_robot_cleanup();
return 0;
}
```