JOURNAL OF NEW RESULTS IN SCIENCE (JNRS)
ISSN: 1304-7981
http://dergipark.gov.tr/jnrs
Research Article

Volume : 7
Issue: 2
Year: 2018
Pages: 35-46

Received: 22.11.2017
Accepted: 08.06.2018

Editors-in-Chief: ***Ebubekir ALTUNTAS***
Area Editor: ***Ahmet FENERCİOĞLU/ Bilge GÖZENER***

ODE (Open Dynamics Engine) Based Walking Control Algorithm for Six Legged Robot

Şahin YILDIRIM[a] , Erdem ARSLAN[a] *

[a]Department of Mechatronics Engineering/ Erciyes University, Kayseri, Turkey
sahiny@erciyes.edu.tr, erdemarslan@erciyes.edu.tr
*Corresponding author

ABSTRACT: In the walking control algorithms, if standard gaits are used, many dynamic effects such as inertial effects, external forces (gravity and friction) are neglected. Furthermore, neglecting dynamic effects does not have much effect on walking performance if masses of robot parts are not too large. On the other hand, as the size of the robot is increased, the masses of the parts will also increase, so dynamic effects will not able to be ignored. Open Dynamics Engine (ODE) is the most popular rigid-body dynamics simulation algorithm in robotic applications. The use of ODE in a real-time model-based control allows the dynamic effects to take into account during the walk. In this study, an ODE based walking control of a six-legged mobile robot was performed and the balancing performance for 5 step linear trajectory of three different gaits (tripod-quadruped-tetrapod) has given in results.
Keywords: *ODE(Open Dynamics Engine), legged robot, walking control, model based control*

1. Introduction

The legged mobile robots do not require a paved surface like wheeled mobile robots. The research done by U.S.A.T.C.D. agency, has shown that 50% of the world's land is not accessible for wheeled mobile robots (U. S. A. T. C. D. Agency 1963). Besides, in many applications (i.e. dangerous material cleaning (Galt, S., et al. 1997), mine detecting (Huang, Qing-Jiu et al. 2003) wheeled mobile robots cannot be used due to the requirement of flat floor.

There are two basic handicaps in the development of legged mobile robots. First-legged mobile robots are less power-efficient than wheeled alternatives (Song, Shin-Min et al. 1984). This problem can be overcome by using lighter and more robust materials in legged robot design. But in this case the total cost of robot will increase because of the quality materials prices. A cheaper solution is to rearrange algorithms that perform legged mobile robot control to work more efficiently.

Once the dynamic model required for the legged mobile robot control is obtained, simplifications are often done so that the model can be easily obtained. In order to provide robust control and to model the robot more easily, linear approaches are developed (ZMP (Vukobratovic M. et al. 1972), SLIP (Blickhan, R. et al. 1993)). Nevertheless, in these models with high nonlinearities because of their structure, the simplification has an adverse effect on the control performance. The use of inverse dynamics in a legged mobile robot control is an ideal control approach (B. Siciliano et al. 2010). But often it is not possible, to

get the inverse dynamics from the legged robot model because of many nonlinear elements in robot model.

By using physics simulators, which have become popular in recent years, the elements (i.e. bodies, joints, constraints) that constitute the robot can be created as exact simulation model.

ODE (Open Dynamics Engine) is an industrial physics simulator that can be used in robot simulation (Yildirim Ş. et al. 2016). Unlike the alternative physics simulators (BULLET, NVIDIA PhysX, Havok, MuJoCo), dynamic models can be implemented in ODE easily. Many mobile robot simulators such as Webots (O. Michel 2004) and Gazebo (N. Koenig et al. 2004) use the ODE physics engine in their software's.

In the legged mobile robot researches, the dynamic forces generated during motion of the robot are often neglected. It is aimed to produce the optimal gaits where the locations of the footholds and the step sequences are changed(Pratihar, Dilip et al. 2002) (Inagaki, Shinkichi, et al 2006)(Erden, Mustafa Suphi et al. 2008). In situations where the robot size is not too large, the dynamic effects appear to have little influence on motion. Consequently, because of the heavy robot parts and heavy loads in practice, it is not negligible dynamic effects during control.

In this study, a walking control algorithm is developed that takes into account the dynamic effects that will occur during motion and produces real-time joint angles required for balanced walking. Afterwards, the tripod-quadruped and pentapod walking patterns used in the standard walking control algorithms were adapted to the real-time and ODE-based walking control algorithm. Herewith, a 5-step linear trajectory motion for 3 different walking modes (tripod, quadruped, pentapod) was experimentally observed and the results were given using graphs and tables.

2. Theory of System

Control systems developed for legged mobile robots generally have two main objectives. The first is to find the solution that consumes the least amount of energy during the movement of the robot (Mei, Yongguo, et al 2004). The second goal is to determine the most balanced walking pattern during the movement of the robot (Kuffner, James J., et al 2002). In order to perform these objectives, it is an obligation to use the inverse dynamic model of the robot. Because of the robot has a lot of nonlinearity in the dynamic model, it is necessary to make some simplifications on model when the inverse dynamics of the robot has obtained. This situation affects the control process adversely, both in terms of balance performance and energy efficiency. By using the ODE physics simulator, it is possible to easily model both the dynamic and the inverse dynamic problem of a legged mobile robot. The work of the ODE algorithm, which operates in discrete time, will be described in detail in this section. In addition, this chapter will describe the proposed walk control algorithm that is necessary for the ODE engine to be used in real time.

2.1. ODE (Open Dynamics Engine)

ODE (Open Dynamics Engine) was developed by Russell Smith in 2007 (Yildirim Ş. et al. 2016). Because the ODE platform runs on a discrete time basis, it is possible to use it in a real-time control algorithm. Three-dimensional vectors are used to define the positions of

the joints, limbs, and constraints in the ODE model. Quaternions are used to define the orientation of the elements. In the ODE platform, the dynamic effects acting on the masses are calculated by Newton-Euler equations. Newton-Euler equations describe the effect between the terms speed and inertia as

$$\overline{f} = m\dot{\overline{v}} \tag{1}$$

$$\overline{\tau} = I\dot{\overline{\omega}} \tag{2}$$

Equation 1 is known as the Newtonian equation and describes the linear relationship between forces and velocities on a body. The velocities and forces given in the equation are vectors representing the three magnitudes in the x, y, and z directions. Equation (2) is known as the Euler equation. Likewise this equation describes the relationship between angular velocities and torque forces acting on the body. If these two equations are combined on a single matrix, the structure given in Eq. 3 is obtained as

$$\begin{bmatrix} m\overline{\delta} & \overline{0} \\ \overline{0} & I \end{bmatrix} \begin{bmatrix} \dot{\overline{v}} \\ \dot{\overline{\omega}} \end{bmatrix} = \begin{bmatrix} \overline{f} \\ \overline{\tau} - \overline{\omega} \times I\overline{\omega} \end{bmatrix} \tag{3}$$

The $\overline{\delta}$ given on the matrix represents the 3x3 unit matrix. Since more than one bodies modeled on the ODE, the equation that is explicit in Equation 4 must be repeated for each body as in Equation 5.

$$m_i\dot{\overline{v}_i} = \overline{g_i} \tag{4}$$

In the equation 4, m is called the mass matrix (6x6), v (6x1) is called the velocity vector, and g (6x1) is called the effort vector. If the expression in Eq. 5 is written for each mass and combined into a single large matrix, the large mass matrix described in Eq. 5 and 6 is obtained.

$$M = diag[m_1, m_2, \ldots m_n],\ \dot{\overline{v}} = \left[\dot{\overline{v}_1}, \dot{\overline{v}_2}, \ldots, \dot{\overline{v}_n}\right],\ \overline{G} = [\overline{g_1}, \overline{g_2}, \ldots, \overline{g_n}] \tag{5}$$

$$M\dot{\overline{v}} = \overline{G} \tag{6}$$

The constraint matrix (Jacobian Matrix) and Lagrange multipliers given in Eq. (7) are used for the constraints caused by joints and contact relations.

$$M\dot{\overline{v}} = \overline{G} + J^T\overline{\lambda} \tag{7}$$

For solving the dynamic system given in Eq. 7, ODE uses first-order Euler integration method. If the equation 7 discretized for Δt time step (Hsu, J. M. & Peters et al. 2014), the multi-body dynamic model with constraints becomes as

$$\begin{bmatrix} \left(\frac{1}{\Delta t}\right) M & -J^T \\ J & 0 \end{bmatrix} \begin{bmatrix} \overline{v}^{n+1} \\ \overline{\lambda} \end{bmatrix} = \begin{bmatrix} \left(\frac{1}{\Delta t}\right) M\overline{v}^n + G \\ J\overline{v}^n \end{bmatrix} \tag{8}$$

In order to solve the forward dynamic problem of the model, it is necessary to find the lagrange multipliers in the equation 8. ODE uses the Projected Gauss Seidel algorithm for computing the Lagrange multipliers (Silcowitz, M. et al. 2011). The matrix in Eq. (8) is an LCP (Linear Complementary Problem) system. Constraint violations may occur in some cases when solving the LCP problem. In order to solve this problem, Russell Smith proposed a parameter named ERP. The $J\overline{v}^n$ term multiply with ERP(Error Reduction Parameter) parameter should be added to the expression given in equation 8 for solving this problem. This process is also known as Baumgarte stabilization (Baumgarte, J. 1972).
To solve the LCP problem, ODE uses Lemke's algorithm. Russell Smith has proposed a parameter named CFM (Constrained Force Margin) to balance the solution of the Lemke algorithms like as ERP parameter. During modeling, the ERP and CFM parameters must be adjusted to match each other.

2.2. ODE based walking control Algorithm

The reason of ODE is used in the gait control algorithm is the need to consider the effects that will occur from accelerations during motion - Explanation of walking control algorithm. The IMU (Inertial Measurement Unit) used on the robot, allows the measure the inclination of robot's trunk in real time. Layered control architecture has been preferred for ODE based walking control algorithm. On the first layer of this control layer there are joint controllers. As there are 18 joints on the robot, 18 joint controllers are used to separate for each joint. Because the ODE operates on a discrete time basis, the joint controller must also be designed in discrete time. In our previous studies, three types of discrete-time (P, PI, PID) controllers were tested for joint controllers. And it has been observed that the fastest and least faulty result is discrete-time P type control. The objective of the joint controller is to position the connected two body according to their defined set value.

In the layered architecture, there is a balance controller on the top of articulation controllers. The purpose of this controller is to calculate the angle of the 18 joints, which will automatically bring the robot into balance when there is any disturbing effect from the outside. These disruptive effects may be a rapid change in the robot's posture or a sudden change in the contact polygon of the robot's.

The goal of the balancing process is to overlap the center of gravity (CoG) and center of polygon. The CoG can be calculated by using the positions and masses of all the parts in robot. Also CoP is a polygon where edge points are created by legs contacting the ground. While the dynamic model is being created on the ODE, the dynamic parameters of the experimental robot have been used. Moreover, this ODE model is operated simultaneously with real robot during control process. Therefore, there is no need for a contact sensing mechanism. Because the contact could be detected directly on the ODE, when the contact occurs or breaks.

The ODE-based walking controller works on the top of the balancing layer. The purpose of this controller is to activate and deactivate the legs of the robot that will use in the stabilizing control, for a predetermined trajectory and gait pattern. During the control, the legs are deactivated from the stabilization process and are taken to the new position and are included in the balance again. Figure 1 shows the block diagram of the ODE based walking control algorithm.

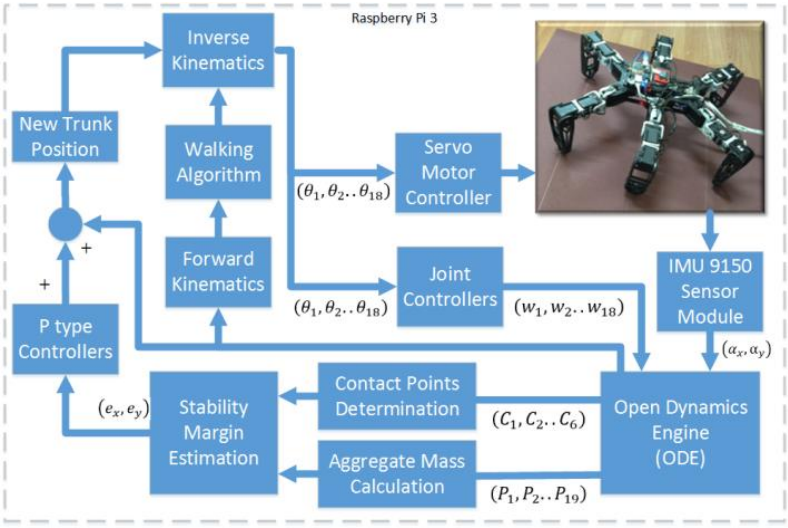

Figure 1. Block Diagram of Walking Control Algorithm

Kinematic calculations are needed between the robot's body and the contact points in order to calculate the position of the legs on the air and on the ground. In the ODE based walking controller which functions as discrete time, the following operations are performed in the infinite loop.

1. Defining the inclination value of ODE model, by using the data from IMU sensor connected to the robot's trunk.
2. Estimation of the legs contacting to ground on ODE model.
3. Calculating the center of gravity (CoG) of the robot using the bodies positions in the ODE model.
4. Calculation of the trunk positioning error which will be used in stabilization process.
5. Multiplying these error values with separate two control coefficient (P type Stabilization Controllers) (X and Y directions).
6. Finding the new body position required for a more balanced posture by adding this difference to the old trunk position.
7. Finding the foothold positions at the beginning of the cycle (with robot's body and current joint angles) (forward kinematic).
8. The updating of these contact points according to the walking sequence and positions (walking control algorithm).
9. Calculation of 18 joint angles to provide balance and walking by using these updated new contact points and new trunk position (inverse kinematics).
10. Sending these 18 joint angle values to joint controllers running on ODE.
11. Sending these 18 joint angle values to servo motor controller on robot.

If these operations are executed in an infinite loop, the control operation in block diagram form in Figure 1 is performed. By means of this walking controller, both the dynamic effects generated during the movement are taken into consideration and the walking is automatically adapted to change environment.

2.3. Stability margins and performance criteria

In legged mobile robot research's, many stabilization margins can be used. In this study a stabilization margin called SSM (Static Stability Margin) were used (De Santos, P. G. et al. 2007). This margin uses the projection of the robot's CoG (Center of Gravity) on the current contact polygon. Normally the SSM is defined by the shortest distance to the edges of the contact polygon. In this study, instead of this margin, the distance between CoG and CoP (Center of Polygon) is defined as positioning error. This margin can be seen on figure 2.

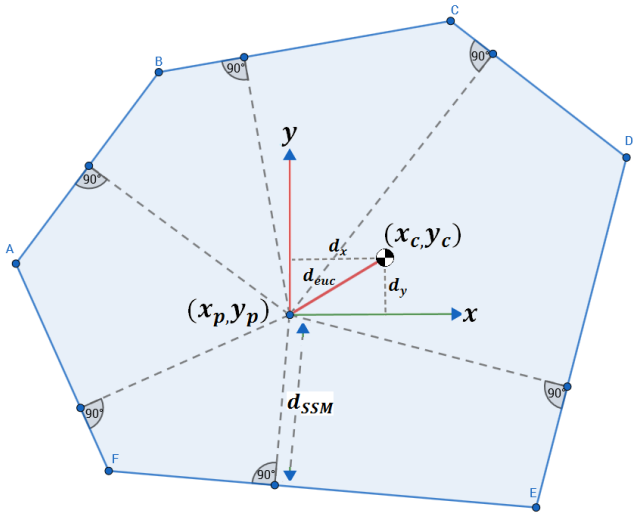

Figure 2. CoG Projection on Contact Polygon

The formulas 9-11 are used to find the center of the polygon shown in Figure 2. With the purpose of forming a closed geometry, the first term is included in the account again when the last term is calculated in sum functions.

$$A = \frac{1}{2}\sum_{i=1}^{n}(x_i.y_{i+1} - x_{i+1}y_i) \tag{9}$$

$$x_p = \frac{1}{6A}\sum_{i=1}^{n}(x_i + x_{i+1})\,(x_i.y_{i+1} - x_{i+1}y_i) \tag{10}$$

$$y_p = \frac{1}{6A}\sum_{i=1}^{n}(y_i + y_{i+1})\,(x_i.y_{i+1} - x_{i+1}y_i) \tag{11}$$

The equations 12-14 are used to find the average center of gravity of the robot.

$$M_T = \sum_{j=1}^{m}(M_j) \tag{12}$$

$$x_c = \frac{1}{M_T}\sum_{j=1}^{m}(x_j.M_j) \tag{13}$$

$$y_c = \frac{1}{M_T}\sum_{j=1}^{m}(y_i.M_i) \tag{14}$$

The positioning errors in the control diagram in Figure 1 are calculated as

$$d_x = x_c - x_p \tag{15}$$

$$d_y = y_c - y_p \tag{16}$$

There is a need for a parameter to express the walking performance, because the objective of this research is to compare the different walking gaits in our proposed control algorithm. Euclidean distance was used as performance function in this study. The performance function is expressed as follows.

$$d_{euc}(t) = \sqrt{d_x(t)^2 + d_y(t)^2} \tag{17}$$

Also a numerical value is required to compare different walking patterns with each other. In order to achieve a numerical value, the performance function is integrated over the time of the motion.

$$e = \int_0^T d_{euc}(t)\, d_t \tag{18}$$

3. Experimental System and Results

The image of experimental six-legged mobile robot is shown in Figure 3. As seen it from figure, the robot has 3 parts for an each leg. These parts are called coxa, tibia and femur respect to the order of connection of the robot trunk. The robot uses 18 Dynamixel Ax-12 servomotors, which will be separate for each joint.

The robot is controlled by a control software developed for the Raspberry Pi 3 SOC on it. During the control process, the robot will sent the data (i.e angular velocity, angular position and torque) to the joints, by using USB2Dynamixel robot controller connected to Raspberry Pi SOC. In addition, there is a MPU-9150 motion processor is used on robot to measure the current inclination value of the robot trunk.

Figure 3. Experimental Six Legged Robot

Each leg has 3 independent servo modules as depicted on Fig 3. There are 18 servo motors on the robot in total. The servo actuators used in experimental six legged robot are "**Dynamixel AX-12**" produced by **Robotis** Inc. The control software we developed on Raspberry Pi was written using the C++ language. Because ODE is an open-source platform, developers regularly publish new versions that increase the performance of the dynamic solver. The ODE version published by Oleh Derevenko on 08.06.2017 was used in this study. In the developed control software, many parameters related to the movement of the robot and the simulation model can be easily adjusted. For example; the P-type kinematic equilibrium coefficient used for walking control how much robot moves on each step (x and y directions), the ERP coefficient of the ODE engine, and the CFM coefficient can be shown.

In this study, the performance of the proposed ODE-based walking control algorithm was shown in three different gaits in which a six-legged mobile robot could walk. The first of these is called tripod walking. In the movement of legged mobile robots, the time between start and the time that the robot reaches at the same posture called one step. Figure 4 shows the leg enumeration on robot for explanation the standard walking gaits. All experiments for this study were repeated for five steps. (for different gaits (tripod, quadruped, pentapod))

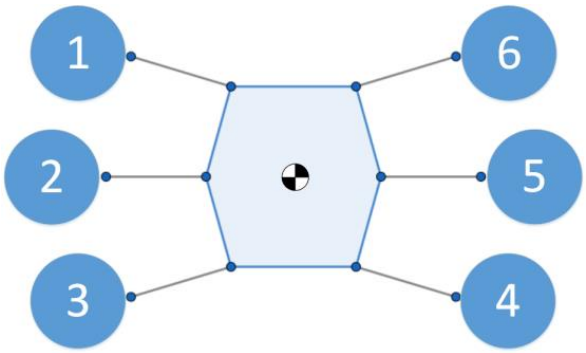

Figure 4. Leg Enumeration on Experimental Robot

Table 1. Standard walking gaits

	Tripod	**Quadruped**	**Pentapod**
1. phase	1-5-3	1-4	1
2. phase	1n-5n-3n	1n-4n	1n
3. phase	6-2-4	2	2
4. phase	6n-2n-4n	2n	2n
5. phase	Recovery	3-6	3
6. phase		3n-6n	3n
7. phase		5	4
8. phase		5n	4n
9. phase		Recovery	5
10. phase			5n
11. phase			6
12. phase			6n
13. phase			Recovery

In Table 1,the movements that the legs made in each phase were given for three different GAITs (tripod-quadruped-pentapod). Normally, in the standard gaits, the last phase is always the recovery phase where the robot's trunk is pulled forward. However, since the algorithm proposed in this study constantly balances the robot, no such phase has been observed in the experiments performed.

Therefore, in terms of speed of motion an important advantage has obtained with the control algorithm we propose. In the experiments, the transition time between phases was given as 0.5 second. The amount of displacement in each step of the robot has given as 3 cm in the + y direction. Figure 5-10 shows the results of the experiments for 3 different gait types. All the numerical results of 3 different gaits are also given in Table 2.

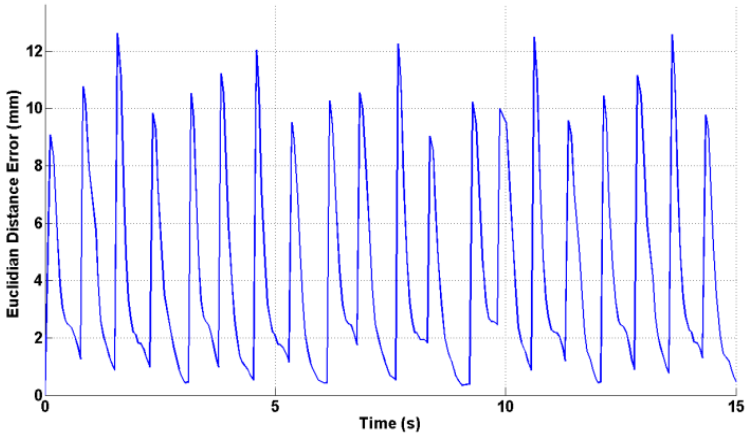

Figure 5. Error Graph of Tripod Walking for 5 Step

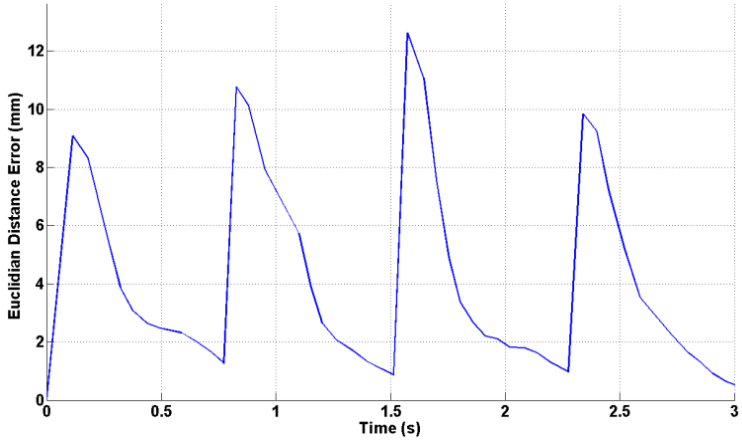

Figure 6. 3 Error Graph of Tripod Walking for a 1 Cycle

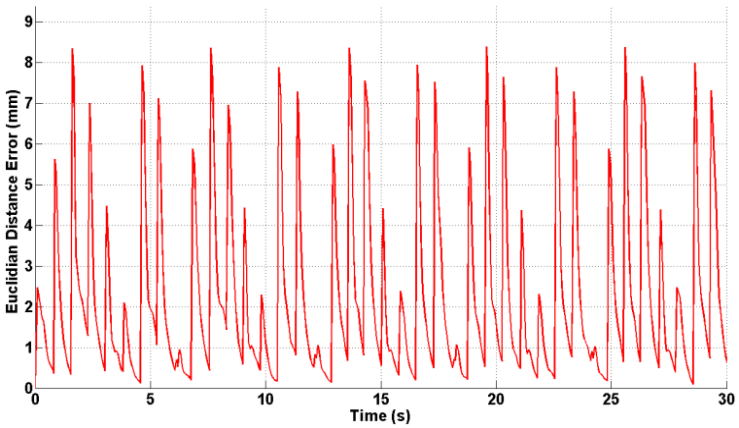

Figure 7. Error Graph of Quadruped Walking for 5 Step

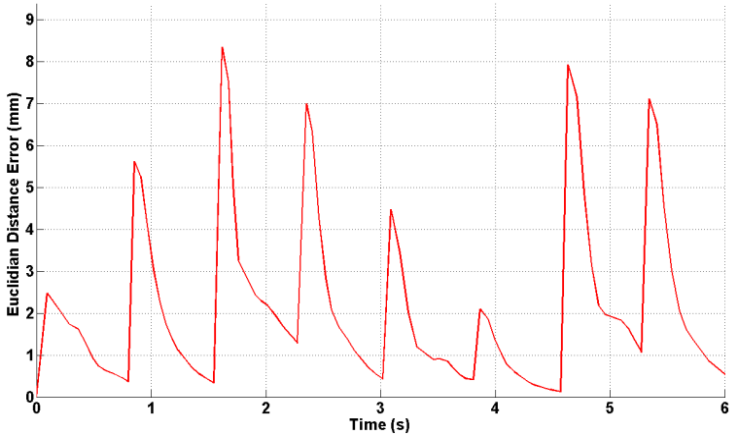

Figure 8. Error Graph of Quadruped Walking for a 1 Cycle

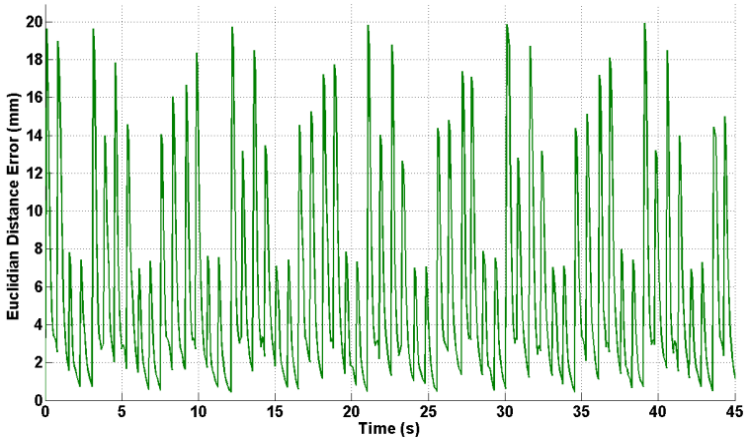

Figure 9. Error Graph of Pentapod Walking for 5 Step

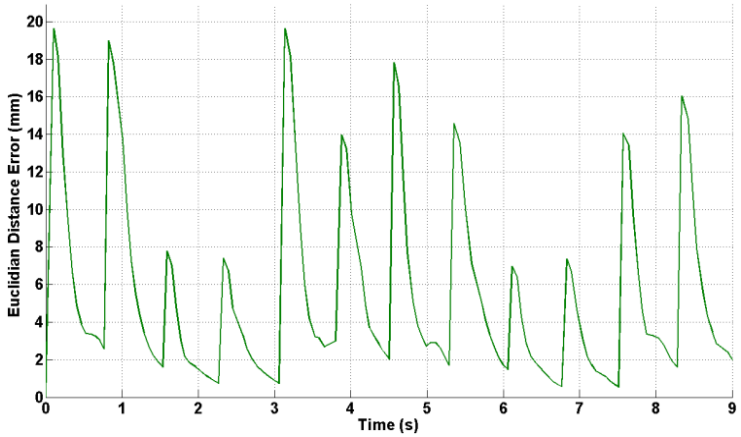

Figure 10. Error Graph of Pentapod Walking for a 1 Cycle

Table 2. Performance results of different walking gaits

Gait Type	Mean Error (mm)	RMS Error (mm)	Area (mm*s)	Max. Error (mm)	Velocity (mm/s)
Tripod	3,62	4,98	60,4	12,62	10
Quadruped	2,02	2,9	64,41	8,38	5
Pentapod	5,1	7,01	241,49	19,92	3,33

4. Conclusion

In this study, an ODE based walking control algorithm was developed, which can take into account all the dynamic effects that would occur during the movement. With this proposed control algorithm, a much more efficient gait control can be achieved compared to standard gait control algorithms. In the experiments, although an active control was performed on the robot, movements similar to those of the same passive walk were observed. In a passive walk, the dynamic forces generated in the system are transferred back to the body in different time. And by doing this, the robot consumes much less energy than needed. In future work, this effect is will be examined to minimize the energy that the robot has consumed.

6. References

B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, Robotics: Modelling, Planning and Control: Springer London, 2010

Baumgarte, J. Stabilization of constraints and integrals of motion in dynamical systems Computer Methods in Applied Mechanics and Engineering, 1972, 1, 1 – 16

Blickhan, R. and Full, R. 1993. Similarity in multilegged locomotion: Bouncing like a monopode. J. Comp. Physiol., A 173:509–517.

De Santos, P. G.; Garcia, E. & Estremera, J. Quadrupedal locomotion: an introduction to the control of four-legged robots Springer Science & Business Media, 2007

Erden, Mustafa Suphi, and Kemal Leblebicioğlu. "Free gait generation with reinforcement learning for a six-legged robot." Robotics and Autonomous Systems 56.3 (2008): 199-212.

Galt, S., et al. (1997). A tele-operated semi-intelligent climbing robot for nuclear applications. Mechatronics and Machine Vision in Practice, 1997. Proceedings., Fourth Annual Conference on, IEEE.

Huang, Qing-Jiu, and Kenzo Nonami. "Humanitarian mine detecting six-legged walking robot and hybrid neuro walking control with position/force control." Mechatronics 13.8 (2003): 773-790

Hsu, J. M. & Peters, S. C. Brugali, D.; Broenink, J. F.; Kroeger, T. & MacDonald, B. A. (Eds.) Extending Open Dynamics Engine for the DARPA Virtual Robotics Challenge Simulation, Modeling, and

Programming for Autonomous Robots: 4th International Conference, SIMPAR 2014, Bergamo, Italy, October 20-23, 2014. Proceedings, Springer International Publishing, 2014, 37-48

Inagaki, Shinkichi, et al. "Wave CPG model for autonomous decentralized multi-legged robot: Gait generation and walking speed control." Robotics and Autonomous Systems 54.2 (2006): 118-126.

Kuffner, James J., et al. "Dynamically-stable motion planning for humanoid robots." Autonomous Robots 12.1 (2002): 105-118.

Mei, Yongguo, et al. "Energy-efficient motion planning for mobile robots." Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on. Vol. 5. IEEE, 2004.

N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), 2004, pp. 2149-2154 vol.3.

O. Michel, "Cyberbotics Ltd. WebotsTM: Professional Mobile Robot Simulation " International Journal of Advanced Robotic Systems, vol. 1, pp. 39-42, 2004.

Pratihar, Dilip Kumar, Kalyanmoy Deb, and Amitabha Ghosh. "Optimal path and gait generations simultaneously of a six-legged robot using a GA-fuzzy approach." Robotics and Autonomous Systems 41.1 (2002): 1-20.

Silcowitz, M.; Niebe, S. & Erleben, K. Richard, P. & Braz, J. (Eds.) Interactive Rigid Body Dynamics Using a Projected Gauss--Seidel Subspace Minimization Method Computer Vision, Imaging and Computer Graphics. Theory and Applications: International Joint Conference, VISIGRAPP 2010, Angers, France, May 17-21, 2010. Revised Selected Papers, Springer Berlin Heidelberg, 2011, 218-229

Song, Shin-Min, et al. "Computer-aided design of a leg for an energy efficient walking machine." Mechanism and machine theory 19.1 (1984): 17-24.

U. S. A. T. C. D. Agency, Logistical Vehicle Off-road Mobility: Final Report: U.S. Army, Transportation Combat Developments Agency, 1963.

Vukobratovic M., Stepanenko Yu.,1972, “On the Stability of Anthromorphic Systems”, Mathematical Biosciences, Vol. 15, pp.1-37

Yildirim Ş., Arslan E., "Estimation of Contact Forces on Real-time Six Legged Mobile Robot with ODE (Open Dynamics Engine)", International Conference on Advances in Mechanical Engineering ICAME 2016, ISTANBUL, TÜRKIYE, 10-13 May 2016, pp.185-190.