electronics

Article

Robot Motion Planning in an Unknown Environment with Danger Space

Hadi Jahanshahi [1], Mohsen Jafarzadeh [2], Naeimeh Najafizadeh Sari [1], Viet-Thanh Pham [3,*], Van Van Huynh [4] and Xuan Quynh Nguyen [5]

[1] Department of Aerospace Engineering, Faculty of New Sciences and Technologies, University of Tehran, Tehran 14395-1561, Iran; hadi_jahanshahi@ut.ac.ir (H.J.); naeimeh.najafi@ut.ac.ir (N.N.S.)
[2] Department of Electrical and Computer Engineering, The University of Texas at Dallas, Richardson, TX 75080, USA; Mohsen.Jafarzadeh@utdallas.edu
[3] Nonlinear Systems and Applications, Faculty of Electrical and Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City, Vietnam
[4] Modeling Evolutionary Algorithms Simulation and Artificial Intelligence, Faculty of Electrical and Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City, Vietnam; huynhvanvan@tdtu.edu.vn
[5] National Council for Science and Technology Policy, Hanoi, Vietnam; Quynhnx@hactech.edu.vn
* Correspondence: phamvietthanh@tdtu.edu.vn

Received: 17 January 2019; Accepted: 7 February 2019; Published: 10 February 2019

Abstract: This paper discusses the real-time optimal path planning of autonomous humanoid robots in unknown environments regarding the absence and presence of the danger space. The danger is defined as an environment which is not an obstacle nor free space and robot are permitted to cross when no free space options are available. In other words, the danger can be defined as the potentially risky areas of the map. For example, mud pits in a wooded area and greasy floor in a factory can be considered as a danger. The synthetic potential field, linguistic method, and Markov decision processes are methods which have been reviewed for path planning in a free-danger unknown environment. The modified Markov decision processes based on the Takagi–Sugeno fuzzy inference system is implemented to reach the target in the presence and absence of the danger space. In the proposed method, the reward function has been calculated without the exact estimation of the distance and shape of the obstacles. Unlike other existing path planning algorithms, the proposed methods can work with noisy data. Additionally, the entire motion planning procedure is fully autonomous. This feature makes the robot able to work in a real situation. The discussed methods ensure the collision avoidance and convergence to the target in an optimal and safe path. An Aldebaran humanoid robot, NAO H25, has been selected to verify the presented methods. The proposed methods require only vision data which can be obtained by only one camera. The experimental results demonstrate the efficiency of the proposed methods.

Keywords: robot path planning; danger space; unknown environment; modified Markov decision processes

1. Introduction

Nowadays robot path planning, as an open problem, are attracting considerable attention. A robot must be capable to work in a large spectrum of environments. In fact, the environment changes dynamically and robots must be able to deal with these changes. Obviously, to navigate in unknown environments, it is not possible to just rely on built-in memories. Bug algorithms are the first path planning algorithms that the robot reaches the target definitively. Due to the real-time performance, these algorithms are very appropriate for the robots. For the first time, Lumelsky and

Skewis used sensors in the Bug algorithms [1]. Some improved Bug algorithms are proposed to reduce the time cost and computational complexity and improve the adaptability and performance. The I-Bug algorithm [2], Point Bug algorithm [3], EgressBug algorithm [4], InsertBug algorithm [5], and H-Bug algorithm [6] can be identified as the improved Bug algorithms.

Because the environment can be changed at any time and only limited information is available at any given time, the strategy of determining the path of the robot must be such that the robot can approach to the target with the least possible information (preferably the position of the robot and the target at any time). González et al. presented a comparative review of intelligent vehicles' motion algorithms in complex environments by considering the safety factor [7]. In another review paper, Ravankar et al. summarized the path planning algorithms for autonomous mobile robots. They focused on the path smoothing techniques which satisfy certain constraints like continuity and safety [8]. From managerial insights, Sarkar et al. have focused on increasing the safety factors and reducing the setting time [9]. Some well-known path-planning techniques like A* [10,11], Dijkstra [12], distance conversion [13,14], potential field [15–19], sampling-based [20,21], and piano stimulation problem [22–24] need more information and sometimes they require a full map. This weakness shows that in unknown environments, point-to-point guidance is necessary. On the other hand, fuzzy logic has been used widely to successfully solve a wide range of problems in various application fields [25–29]. As a more potent tool, Zavlangas et al. proposed a fuzzy-based algorithm to navigate the omnidirectional mobile robots [30]. The proposed strategy considers only the nearest obstacle to decide on the next robot's move. Although this method is real-time and seems efficient, these parameters are not provided for a humanoid robot with on camera. In other words, this method can be used for robots with omnidirectional range sensors. An effective approach to navigate the wheeled mobile robots has been presented by Al Yahmedi and Fatmi [31]. Issues of individual behavior design and action coordination of the behaviors were addressed using fuzzy logic. This resulted in saving the time and computational resources. The research involves 14 sensors to find the positions of all obstacles around the robot. Thus, this method cannot be implemented on the most humanoid robots. In another work, Iancu et al. presented a fuzzy reasoning method of Takagi–Sugeno type controller to navigate a two-wheeled autonomous robot [32]. This mobile robot is equipped with a sensorial system which contains seven radial sectors. Most humanoid robots do not have such a system of sensors and so this method cannot be implemented on them.

A path-planning algorithm for the humanoid robots is proposed by Michel et al. [33]. They used an external camera that provides a top view of the environment for the robot to obtain information of the position of the obstacles. Their method is not applicable in most situations because it is impossible to use a camera with a global view of the robot work sites. Besides, Nakhaei and Lamiraux used the online 3D mapping and combined it with path planning task. They used a roadmap-based method for motion planning because the dimension of configuration space is high. This algorithm was implemented on HRP2 [34]. Their method is not efficient because it needs exact stereo vision and a lot of time to find a path in each step. Furthermore, Sabe et al. presented a method for path planning and obstacle avoidance for the humanoid robot Quest for cuRIOsity (QRIO). This algorithm allows the robot to move in a home-like environment [35]. The A* algorithm was used in this method, which requires high processing time. Their method seems effective, but it needs stereo vision and high computational processes. As a result, it cannot be applied in most conditions. Another path-planning project on HRP-2 humanoid robot is done by Michel et al. [36]. This method used several cameras to produce the maps. This method is inefficient because of the constraints in embedding the cameras. Meanwhile, Chestnutt et al. implemented best-first search and A* algorithms for footstep path planning of H7 humanoid robot [37]. Both of them need stereo vision and high computational processes. In another work, Okada et al. proposed another method for path planning of a humanoid robot [38]. In this method, robot and obstacles were modeled with cylinders and vision helped eliminate the floor from the decision. This method may encounter with a conflicting problem when the robot confronts a big obstacle at the start point. In this situation, the robot could not be able to detect the floor and may miss the path.

One of the most important tasks of mobile robots is to move in environments that include danger space or sensitive areas. This should be considered in the path planning algorithms. The real-world robots' workspaces such as fire in rescue mission and landmines usually includes numerous danger sources. To navigate the robot in a safe and optimal path in such an environment, Zhang et al. introduced a multi-objective path planning algorithm based on particle swarm optimization [39]. In a similar paper, Purcaru et al. proposed a new optimal path planning algorithm in which a hybrid of the gravitational search algorithm and the particle swarm optimization algorithm is implemented [40]. In this algorithm, the robot tries to avoid collisions with danger spaces and obstacles, in addition to moving in the shortest possible path from start point to the target. Zhang et al. suggested an improved A* path planning algorithm to create a smooth and safe path with regard to the potentially risky areas of the map [41]. The robot path planning using the artificial potential field procedure is one of the most popular path planning methods. By implementing the artificial potential field method, Matoui et al. proposed a path planning algorithm to push the robots far from the danger space in unknown environment [18].

As can be seen from the above study, an appropriate and efficient method for path planning of a humanoid robot in an unknown environment is still not proposed. Considering the identified research gap, four methods including synthetic potential field method, Linguistic method, Markov decision processes, and fuzzy Markov decision processes are studied. In this paper, at first, the color model is discussed. After that, Synthetic potential field, linguistic method, Markov decision processes, and fuzzy Markov decision processes are introduced and implemented for path-planning in unknown environments. Finally, the path planning in an environment regarding the presence of a danger space is discussed.

2. Robot Path Planning Using Vision Sensors

Sensor-based path planning uses three different sensors including occupancy sensor, distance sensor, and vision sensor. The occupancy sensor usually extends the path and provides the least information to the robot. For this reason, methods that rely solely on these sensors are obsolete. The distances sensors provide good data for path planning, but the extracted data does not provide the ability to identify the danger zones for the robot. The vision sensors are the best sensors in the robot path planning, given the information they provide.

2.1. The Color Models

Differences in the frequency and wavelengths differentiate colors. Therefore, a way to display pixels is storing the frequency vector and the corresponding light intensity. Unfortunately, current technology does not allow such sensors to be build. As a result, different methods have been developed to measure and store pixels. The most famous color model is called the Red Green Blue (RGB) model, in which the intensity of the three red, green, and blue colors is stored. Yellow, turquoise, and purple are three subclasses produced from the combination of each pair of these original colors (Figure 1). The white color refers to the presence of all three original colors and the black color refers to the absence of any of these three original colors. Typically, the color intensity is shown by integers between 0 and 255. To obtain light intensity (regardless of color), it is enough to measure the mean intensity of the three main colors. The human eye has cells that are sensitive to the frequency of red, green, and blue lights and therefore functions similar to the RGB model.

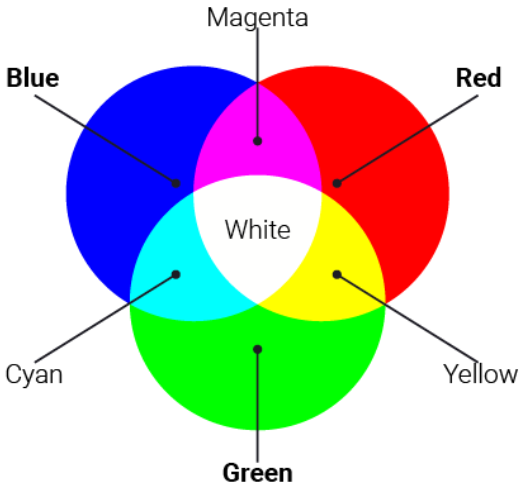

Figure 1. The combination of red (R), blue (B), and green (G) colors in the RGB color model.

Robots equipped with vision sensors use different models depending on the type of sensor used. As a result, in the first step, there should be a tool for transforming their color model. For example, the Nao humanoid robot uses the 422 YUV color model (Figure 2). The Luminance (Y), blue–luminance (U), and red–luminance (V) refers to a system which defines color with one luminance value and two chrominance values.

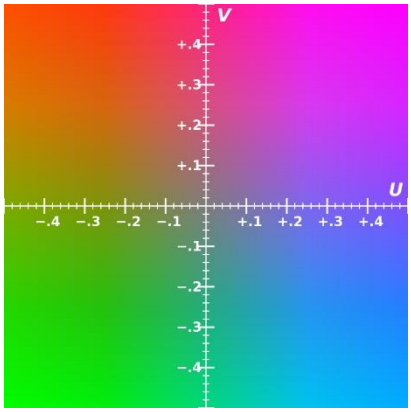

Figure 2. YUV color model.

Using Equation (1), the 422 YUV can be converted to the RGB.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.13983 \\ 1 & -0.39465 & -0.58060 \\ 1 & 2.03211 & 0 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix} \quad (1)$$

2.2. Low-Pass Filter

Due to the electromagnetic nature of light, the noise in the sensors is normal. Due to the continuity of the bodies, there is also an approximation continuity of color. As a result, the presence of a high-frequency signal (the color difference of one pixel relative to its neighbors) is likely to be noise. As a result, the noise should be deleted by using the low-pass filter.

2.3. Segmentation and Mode Filter

As mentioned, each pixel represents three intensity levels of red, green, and blue which varies between 0 and 255. Although these data are needed to display the image, they do not play a role in

the path planning. A robot only needs to know the type of pixel to determine its path. To do this, it must be determined that each pixel indicates which obstacle, danger, free, or target spaces. The conversion of the image from the color space to the environmental space is called segmentation. Various ways to segment an image are suggested. Selecting the appropriate segmentation method depends on the location of the robot. All segmentation techniques can have an error (noise). In order to remove this noise, the image must be passed through the Mode filter.

2.4. Expansion

During the recording process, the parts of the outer boundary of the obstacles may be recorded in the form of other spaces (usually free space). As a result, to avoid collisions, obstacles are slightly widened. Also, the expansion of the obstacles increases the effect of the small obstacles. Figure 3 shows the image expansion.

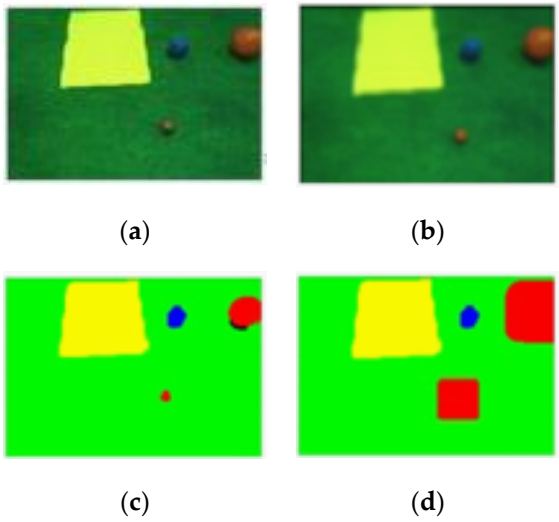

Figure 3. Image expansion to enhance the obstacle effect (**a**) main image; (**b**) filtered image; (**c**) segmented and filtered image; (**d**) expanded image.

2.5. Schematic Structure of Vision System

The humanoid robot implemented in this paper uses a camera to collect environmental information. After capturing an image by the camera, the image passes through a filter to obviate the concomitant noise. The next step is the segmentation of the image. After image segmentation, the image passes through a mode filter to remove the effect of noise generated during the segmentation. Finally, the dilation process is applied to produce the final image. The structure of the vision system can be simplified as Figure 4.

Figure 4. Schematic structure of the vision system.

3. Path Planning in the Absence of Danger Space

Here, four approaches including synthetic potential field method, linguistic method, Markov decision processes, and fuzzy Markov decision processes are reviewed and implemented on the Aldebaran humanoid robot–Nao H25 V4.

3.1. Synthetic Potential Field Method

A pair of electrical charges exert a force on each other as follows [42]:

$$\vec{F} = k\frac{q_1 q_2}{|r|^2}\vec{e_r} \tag{2}$$

In this equation, k is a constant, q_1 and q_2 are electrical charges, r is the distance between them, and $\vec{e_r}$ is the unit vector connecting these two electrical charges. It is supposed that robot and obstacles carry negative charges, while the target has a positive one. As the result, obstacles repulse and the target attract the robot. From electrostatic laws, it is concluded that one positive and N negatives charges exert a force to a negative charge as Equation (3):

$$\vec{F} = \vec{F_a} + \sum_{k=1}^{N} \vec{F_d}(k) \tag{3}$$

where F_a and F_d are attraction and repulsion forces, respectively. Additionally, k represents the number of obstacles. By confining each cell of meshed space to the environmental space, Equation (3) is rewritten as Equation (4).

$$\vec{F} = \vec{F_a} + \sum_{k=1}^{N} \vec{F_d}(i,j) \tag{4}$$

where (i,j) represents the cell's position. Additionally, n represents the number of cells in a row and $k = nj + i$. Forces F_a and F_d are calculable from the vector decomposition along the main x and y axes as below:

$$\vec{F_a} = F_{ax}\boldsymbol{i} + F_{ay}\boldsymbol{j} \tag{5}$$

$$\vec{F_d}(i,j) = F_{dx}(i,j)\boldsymbol{i} + F_{dy}(i,j)\boldsymbol{j} \tag{6}$$

The respective components are as below:

$$F_{ax} = \vec{F_a}.\frac{\vec{x}_{goal}}{|r_{goal}|} \tag{7}$$

$$F_{ay} = \vec{F_a}.\frac{\vec{y}_{goal}}{|r_{goal}|} \tag{8}$$

$$F_{dx}(i,j) = \vec{F_d}(i,j).\frac{\vec{x}(i,j)}{|r_{(i,j)}|} \tag{9}$$

$$F_{dy}(i,j) = \vec{F_d}(i,j).\frac{\vec{y}(i,j)}{|r_{(i,j)}|} \tag{10}$$

By inserting Equations (7) and (8) into Equation (5), and inserting Equations (9) and (10) into Equation (6), Equations (11) and (12) can be obtained as follows:

$$\vec{F_a} = \vec{F_a}.\frac{\vec{x}_{goal}}{|r_{goal}|}\boldsymbol{i} + \vec{F_a}.\frac{\vec{y}_{goal}}{|r_{goal}|}\boldsymbol{j} \tag{11}$$

$$\vec{F}_d(i,j) = \vec{F}_d(i,j).\frac{\vec{x}_{goal}}{|r_{goal}|}\boldsymbol{i} + \vec{F}_d(i,j).\frac{\vec{y}_{goal}}{|r_{goal}|}\boldsymbol{j} \quad (12)$$

Considering Equations (11) and (12), Equation (4) changes as follows:

$$\vec{F} = \vec{F}_x + \vec{F}_y \quad (13)$$

where

$$\vec{F}_x = \left(\vec{F}_a.\frac{\vec{x}_{goal}}{|r_{goal}|} + \sum_{k=1}^{N}\vec{F}_d(i,j).\frac{\vec{x}(i,j)}{|r_{(i,j)}|}\right)\boldsymbol{i} \quad (14)$$

$$\vec{F}_y = \left(\vec{F}_a.\frac{\vec{y}_{goal}}{|r_{goal}|} + \sum_{k=1}^{N}\vec{F}_d(i,j).\frac{\vec{y}(i,j)}{|r_{(i,j)}|}\right)\boldsymbol{j} \quad (15)$$

Since all obstacles are the same (attributed as obstacle space), Equation (2) can be rewritten for obstacles as:

$$\vec{F}_d(i,j) = -k_d\frac{1}{(|r_{(i,j)}|)^2}\vec{e}_r \quad (16)$$

Considering the target point, it is possible to rewrite the attraction equation as follows:

$$\vec{F}_a = k_a\frac{1}{(|r_{goal}|)^2}\vec{e}_r \quad (17)$$

By inserting Equations (17) and (16) into Equation (14) and Equation (15), Equations (18) and (19) can be obtained as follows:

$$\vec{F}_x = \left(k_a.\frac{\vec{x}_{goal}}{(|r_{goal}|)^3} - k_d\sum_{k=1}^{N}\frac{\vec{x}(i,j)}{(|r_{(i,j)}|)^3}\right)\boldsymbol{i} \quad (18)$$

$$\vec{F}_y = \left(k_a.\frac{\vec{y}_{goal}}{(|r_{goal}|)^3} - k_d\sum_{k=1}^{N}\frac{\vec{y}(i,j)}{(|r_{(i,j)}|)^3}\right)\boldsymbol{j} \quad (19)$$

Each cell (except the target point) can share obstacle and free space beside some uncertainty. So, to fuzzify these equations, the magnitude of the repulsive force is multiplied to the membership function of obstacle space (μ) [43]. Thus, Equations (20) and (21) can be obtained as:

$$\vec{F}_x = \left(k_a.\frac{\vec{x}_{goal}}{(|r_{goal}|)^3} - \mu k_d\sum_{k=1}^{N}\frac{\vec{x}(i,j)}{(|r_{(i,j)}|)^3}\right)\boldsymbol{i} \quad (20)$$

$$\vec{F}_y = \left(k_a . \frac{\vec{y}_{goal}}{(|r_{goal}|)^3} - \mu k_d \sum_{k=1}^{N} \frac{\vec{y}(i,j)}{(|r_{(i,j)}|)^3} \right) \boldsymbol{j} \quad (21)$$

To reach the target, the robot must move in the direction of the force. In other words, the robot angle is calculated using the following equation:

$$\phi = arctan2\left(\vec{F}_y, \vec{F}_x\right) \quad (22)$$

where "arctan2" is defined as Equation (23):

$$arctan2\left(\vec{F}_y, \vec{F}_x\right) = \begin{cases} arctan\left(\frac{y}{x}\right) x > 0 \\ arctan\left(\frac{y}{x}\right) + \pi y \geq 0; x < 0 \\ arctan\left(\frac{y}{x}\right) - \pi y < 0; x < 0 \\ \frac{\pi}{2} y > 0; x = 0 \\ \frac{-\pi}{2} y < 0; x = 0 \\ undefiney = 0; x = 0 \end{cases} \quad (23)$$

3.1.1. The Rectifier

In close proximity of target point, the denominator of fractions in (20) and (21) tend to zero and accordingly, attraction takes large magnitudes. This matter results in the ineffectiveness of repulsive force from obstacles. To overcome this liability, it is proposed to add integer 1 to the denominator of the relevant fraction.

$$\begin{aligned} \phi = arctan2(& k_a . \frac{\vec{y}_{goal}}{(|r_{goal}|)^3 + 1} \\ & - \mu k_d \sum_{k=1}^{N} \frac{\vec{y}(i,j)}{(|r_{(i,j)}|)^3}, k_a . \frac{\vec{x}_{goal}}{(|r_{goal}|)^3 + 1} \\ & - \mu k_d \sum_{k=1}^{N} \frac{\vec{x}(i,j)}{(|r_{(i,j)}|)^3}) \end{aligned} \quad (24)$$

Figure 5 shows the effect of this change.

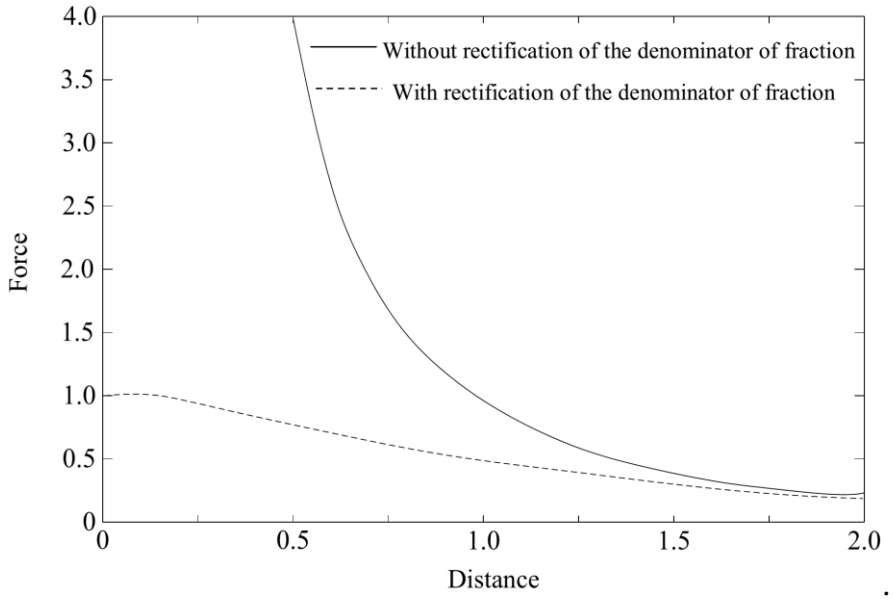

Figure 5. Attraction force with and without the rectifier.

3.1.2. Result of Synthetic Potential Field Method

As previously mentioned, to evaluate the proposed method, Aldebaran humanoid robot–NAO H25 V4 is used. The construction of employed humanoid robot including implemented actuators and sensors are described [44]. Additionally, the control mechanism of this robot is presented in [45]. The software architecture was developed using Aldebaran's NaoQi as a framework and an extended code in C++. In this way, Kubuntu 12.0.4 and Open CV 2.3.1 writing program in C++ in Qt creators is used. In Figure 6, at the beginning of the process, the target is considered a virtual point. In the first step, the robot does not see any obstacles and decides to move directly to the target. By observing the first obstacle by the robot, a distraction force will be added. The resultant force causes the robot to move between the two obstacles.

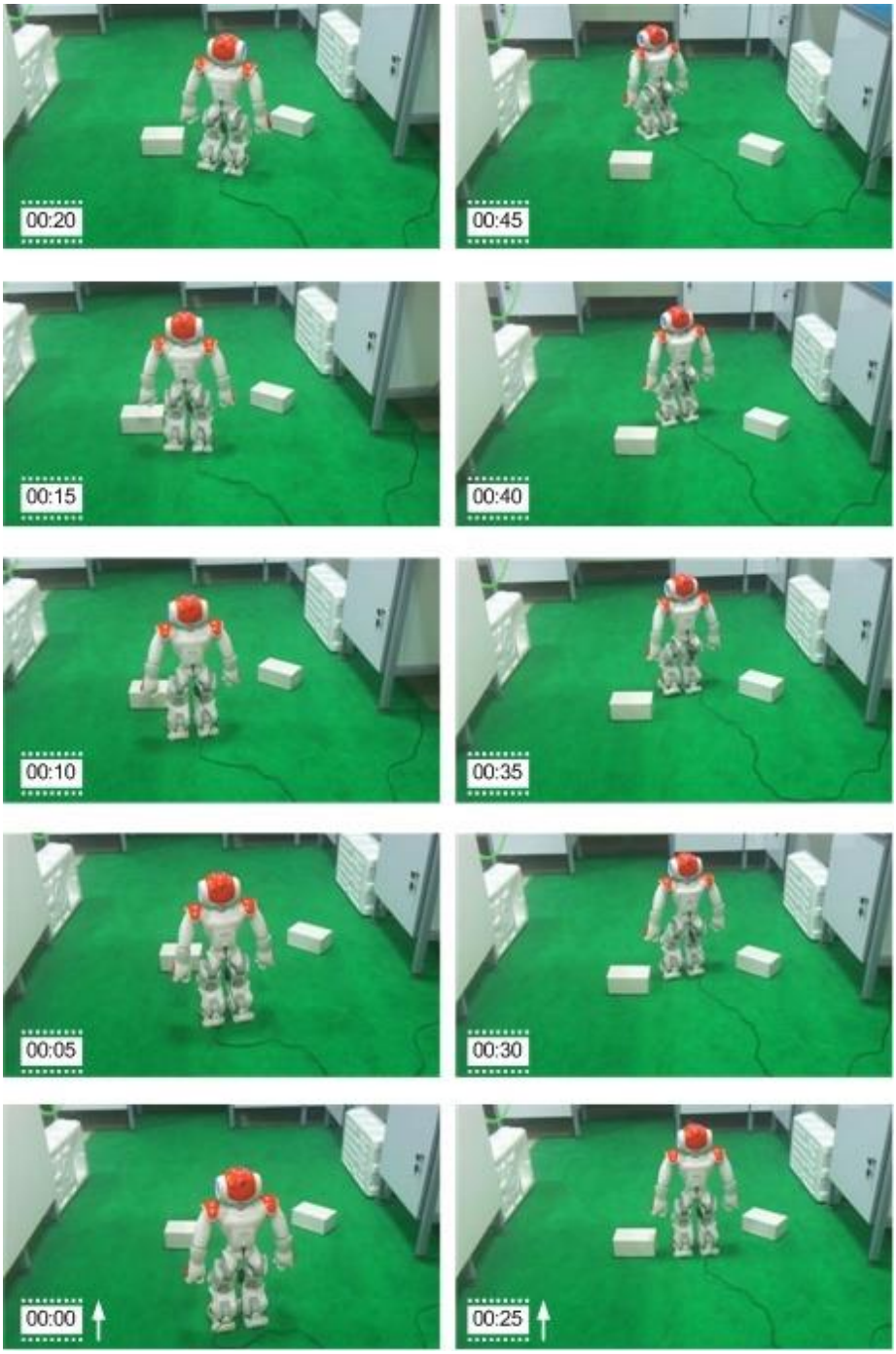

Figure 6. The path traversed by the Nao humanoid robot using the fuzzy synthetic potential field method (see the video of the robot's movement).

3.2. Linguistic Method

While the linguistic method has a root in the natural potential field, it tries to compute the field by linguistic rules, instead of deterministic relations. The intensity of natural potential force is proportional with square of distance. Also, the intensity of the synthetic potential field must be a descending function of distance. Regarding the dimensions of the Nao and the height of its camera, the taken image is divided into 25 cells. Tables 1 and 2 summarize the rules of the calculating forces of the obstacles and the target point. The variables are Positive (P), Negative (N), Small (S), Zero (Z), Very (V), Medium (M), and Big (B).

Table 1. The rules of obstacles' force in the direction of the *x*- and *y-axis*.

		Axis *x*					**Axis *y***				
j	***i***	**1**	**2**	**3**	**4**	**5**	**1**	**2**	**3**	**4**	**5**
1		VSN	VSN	Z	VSP	VSP	VSP	VSP	VSN	VSN	VSN
2		VSN	VSN	Z	VSP	VSP	VSN	SN	SN	SN	VSN
3		SP	SN	Z	SP	SP	SN	MN	MN	MN	SN
4		MN	MN	Z	MP	MP	MN	BN	VBN	BN	MN
5		BN	VBN	Z	VBP	BP	MN	VBN	VBN	VBN	MN

Table 2. The rules of the target's force in the direction of the *x*- and *y*-axis.

		Axis *x*					**Axis *y***				
j	***i***	**1**	**2**	**3**	**4**	**5**	**1**	**2**	**3**	**4**	**5**
1		VSP	VSP	Z	VSN	VSN	VSP	VSP	VSP	VSP	VSP
2		VSP	VSP	Z	VSN	VSN	VSP	SP	SP	SP	VSP
3		SP	SP	Z	SN	SN	SP	MP	MP	MP	SP
4		MP	MP	Z	MN	MN	MP	BP	VBP	BP	MP
5		BP	VBP	Z	VBN	BN	MP	VBP	VBP	VBP	BP

When the target is seen by the robot, through the fuzzification, a non-zero magnitude is assigned to the cells in which the target is located. This obviously attracts robot to the target. On the other hand, if the robot fails to view the target, it receives a virtual repulsive force and gives up to approach the target. To overcome this problem, it is assumed that a virtual target is located in the closest cell to the main target (Figure 7).

Regarding *n* and *m*, there are totally 4 nm fuzzy rules that determine the level of the output. In the natural potential field, the force exerted on a charged body is the sum of the single forces issued by other charged bodies (superposition principle). As a guide, this matter can be implemented for the synthetic potential field method. In this method, the superposition is equivalent to the weighted average defuzzification that can be defined as follows:

$$f_x = \frac{\sum_{i=1}^{N} P_{obstacle}^{i} \hat{f}_{rx}^{i} + P_{target}^{i} \hat{f}_{ax}^{i}}{\sum_{i=1}^{N} P_{obstacle}^{i} + P_{target}^{i}} \quad (25)$$

$$f_y = \frac{\sum_{i=1}^{N} P_{obstacle}^{i} \hat{f}_{ry}^{i} + P_{target}^{i} \hat{f}_{ay}^{i}}{\sum_{i=1}^{N} P_{obstacle}^{i} + P_{target}^{i}} \quad (26)$$

where $P^i_{obstacle}$ is the probability of the presence of the obstacle, P^i_{target} is the probability of the presence of the target, and f_x and f_y are the force components along the x- and y-axis, respectively.

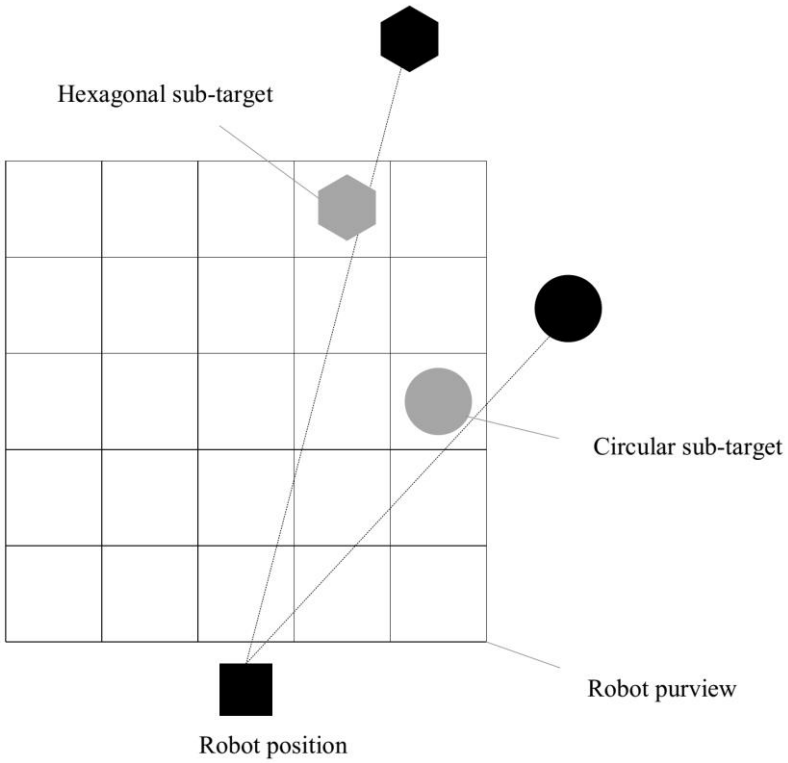

Figure 7. Positions of robot and sub-targets in the purview of the robot.

3.2.1. Simplification

In the proposed method, the robot moves in the direction of the field. So simply knowing the direction of the field is enough. As can be seen in Equations (25) and (26), denominators of fractions are the same and it is possible to multiply both equations into the common denominator to simplify them without changing directions of the forces.

$$f_x^{'} = \sum_{i=1}^{N} P^i_{obstacle} \hat{f}^{\,i}_{rx} + P^i_{target} \hat{f}^{\,i}_{ax} \quad (27)$$

$$f_y^{'} = \sum_{i=1}^{N} P^i_{obstacle} \hat{f}^{\,i}_{ry} + P^i_{target} \hat{f}^{\,i}_{ay} \quad (28)$$

The direction of the force can be determined as follows:

$$\phi = arctan2(f_y', f_x') \quad (29)$$

3.2.2. Result of Linguistic Method

Figure 8 illustrates the path traversed by the Nao robot through the use of the linguistic method. At the beginning of the process, the robot is unable to see the target. So, the robot uses a sub-target and thus tries to approach the approximated target position. The robot moves between obstacles and approaches the sub-target. After the target is identified by the robot, the robot directly moves to the target without colliding with obstacles.

Figure 8. The path traversed by the Nao humanoid robot using the linguistic method (see the video of the robot's movement).

3.3. Markov Decision Processes

Markov decision processes presented a mathematical framework for decision-making modeling in situations that the outcomes are random and out of control [46]. The Markov decision processes is the generalized form of Markov chains. In other words, Markov decision process is a discrete time stochastic control process. So, in each time step, the state of the process is s and the decision maker chooses an action form the possible actions. After that, the process randomly moves to the next state s', and reward R is given to the decision maker [47]. Therefore, the probability that the process will go to a certain state is a function of the chosen action. This means that the state s' depends on the state s and the action of the decision-maker a. This is while a and s are independent of all former actions and states. In other words, moving from a state to another in Markov decision

processes has Markov property [48–50]. The main problem in Markov decision processes is finding a function π that specifies the action $\pi(s)$ that the decision maker will choose when in state s. Actually, the goal is to find a policy π which will maximize some cumulative function of the random rewards. Additionally, the value function represents the magnitude of expected rewards which a system receives by working from state s and following the policy. Therefore, each policy leads to a value function as follows:

$$V^{\pi}(s_0) = E[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots |\pi] = E[\sum_{t=0}^{N} \gamma^t R(s_t)|\pi] \quad (30)$$

This equation can be rewritten as below:

$$V^{\pi}(s_0) = E[R(s_0) + \gamma(R(s_1) + \gamma R(s_2) + \cdots |\pi] \quad (31)$$

Equation (31) is named after Bellman and is abbreviated to:

$$V^{\pi}(s) = E\left[R(s) + \gamma \sum_{s'} P(s,a,s')V^{\pi}(s')\right] \quad (32)$$

By displaying the optimum policy and the optimum value function with π^* and V^*, respectively, Equations (33) and (34) can be obtained as:

$$V^*(s) = R(s) + max_a \gamma \sum_{s'} P(s,a,s')V^*(s') \quad (33)$$

$$\pi^*(s) = argmax_a \sum_{s'} P(s,a,s')V^*(s') \quad (34)$$

By assuming n states, there will be n equations which constitute a solvable system of equations.

3.3.1. Path Planning

Determining the direction of movement of the robot in each step using the Markov decision processes is possible. To do so, a collision with an obstacle will result in a negative reward. However, achieving the target point will embody a positive reward. As the result, in order to avoid the negative reward, the robot may prefer to stay motionless in some situations, such as near an obstacle. To overcome this problem, a small negative reward (compared to the big negative reward of obstacles) is assigned to the free space.

The calculation of the reward is based on the observation of the target. If the robot can see the target, the robot will rely on information obtained from the image. But if the robot cannot see the target, in addition to the information obtained from the image, it needs its own coordinates and the target coordinates in order to create a sub-target.

A sub-target, that can be defined as a virtual target, could guide the robot toward the original target. Figure 9 shows how to calculate the state of the sub-targets. As seen in this figure, if the black hexagon is selected as the target, the gray hexagon is defined as the sub-target. Similarly, if the black circle and the black triangle are considered as the target, the gray circle, and the gray triangle are defined as the sub-target, respectively.

In situations where the robot is unable to see the obstacle, the free space has $-w_1$ point, the obstacle has $-w_2$ point, and the sub-target has +1 point. Therefore, the reward function could be calculated as follows:

$$R(i,j) = P_{sub-target}(i,j) + w_1 P_{freespace}(i,j) + w_2 P_{obstacle}(i,j) \quad (35$$

)

On the other hand, in situations where the robot can see the obstacle, the target has +1, the free space has $-w_1$ point, and the obstacle has $-w_2$ point. So, the reward function could be calculated as follows:

$$R(i,j) = P_{target}(i,j) + w_1 P_{freespace}(i,j) + w_2 P_{obstacle}(i,j) \quad (36)$$

In some situations, as in Figure 9, the target is out of the robot field of view. Therefore, instead of the main target, a sub-target is utilized. Figure 10 shows the probable movement function of the robot, assuming that it moves forward.

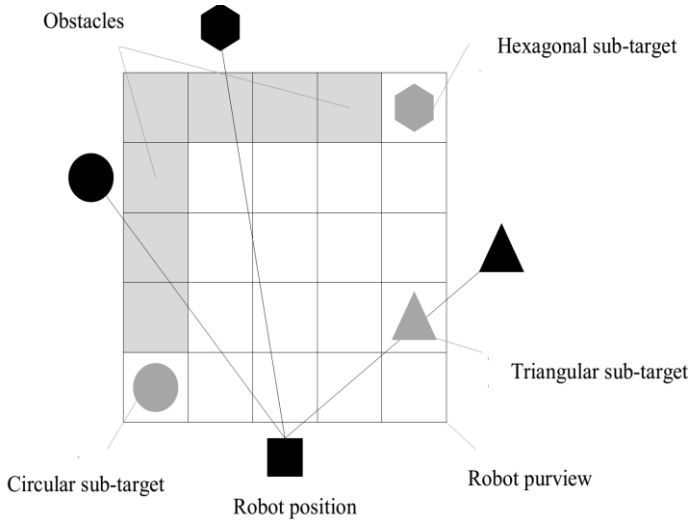

Figure 9. The procedure for assigning sub-targets for the main target.

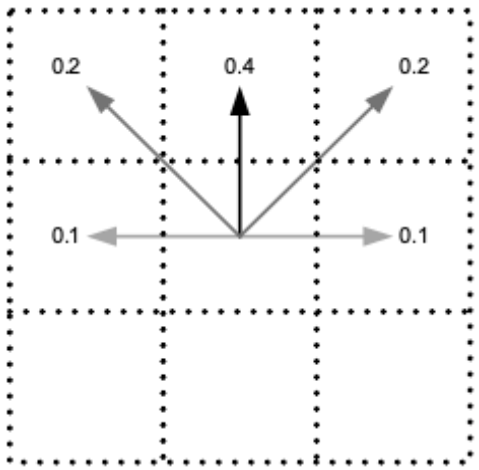

Figure 10. The probability of moving to other states based on selecting forward movement.

The Bellman equation (Equation (33)) is nonlinear and hard to solve. In this case, to obtain the optimal value function without directly solving the Bellman equation, the Algorithm 1 is used:

Algorithm 1 Optimal value function.
Input: Reward function R(s)
Output: Value function V(s)
Begin

$$\forall s V(s) \rightarrow 0$$

repeat
for all the *s* **do**

$$R(s) + max_a \gamma \sum P(s,a,s') V(s') \rightarrow B(s)$$

end

$$B(s) \rightarrow V(s)$$

until $V(s)$ converges;
end

Markov decision processes, always, propose an optimal path based on the current state. Due to the fact that the robot does not always have a full view of its environment, the optimal path is not always the best choice. For example, when the robot moves along a wall and cannot see the target, the robot tries to approach directly to the target. In this situation, the robot may have a severe collision with the wall. A rectifier can unravel this problem by informing the robot from lateral obstacles and preventing a collision in the next steps.

3.3.2. Results of Markov Decision Processes

Figures 11 and 12 are the results of the implementation of the Markov decision processes on the Nao humanoid robot. In these two figures, the arrangement of obstacles is different. As can be seen from these figures and the videos associated with these figures, the robot successfully approaches the target without colliding with obstacles.

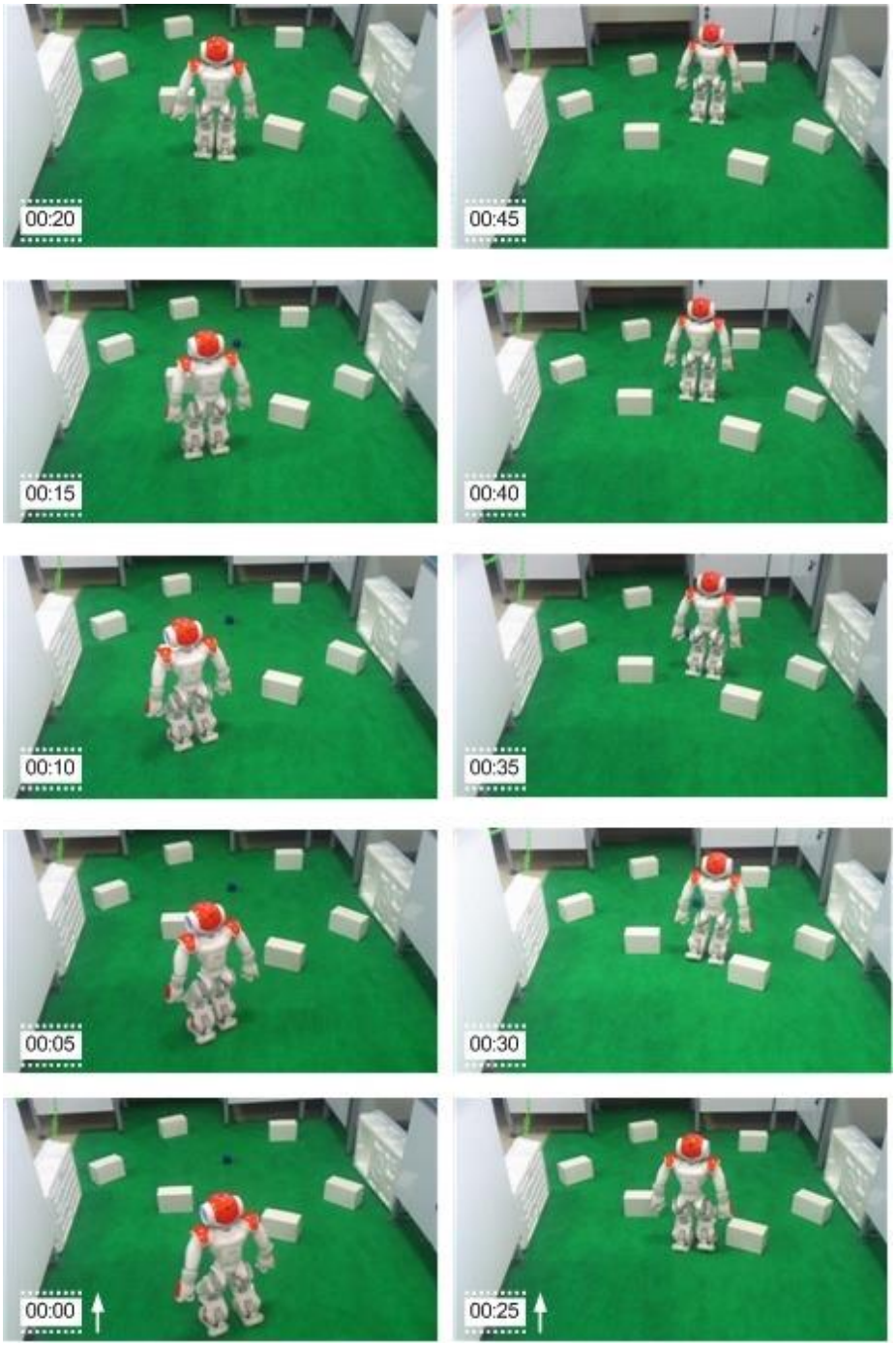

Figure 11. The path traversed by the Nao humanoid robot using the Markov decision processes in the first sample environment (see the video of the robot's movement).

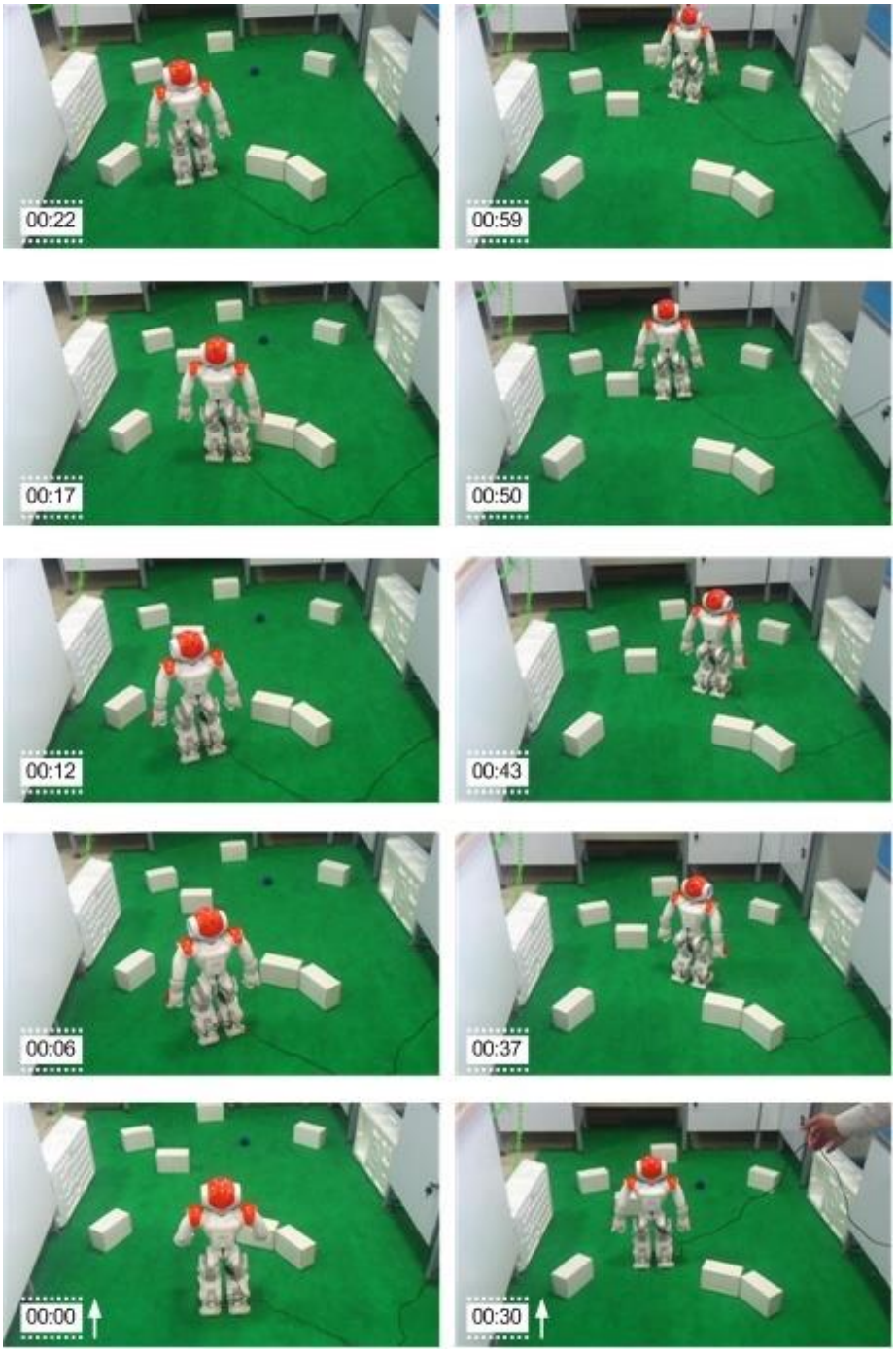

Figure 12. The path traversed by the Nao humanoid robot using the Markov decision processes in the second sample environment (see the video of the robot's movement).

3.4. Fuzzy Markov Decision Processes

As previously stated, the policy of the Markov decision processes is the choice of an action that leads to the highest reward. Here the classic decision-making framework is being modified. For this purpose, as the first step, the value function is evaluated. After that, a fuzzy system determines the action based on the function. According to the reward, the value function takes different intervals in different steps. This is while the inputs of the fuzzy inference engine are fuzzy sums of values 0 to 1. So, the cost function must be normalized in the first step. Thus, the normalized value function can be obtained as follows:

$$V^{new}(i,j) = \frac{V^*(i,j) - b}{a - b}; a = max_{(i,j)}V^*(i,j); b = min_{(i,j)}V^*(i,j) \quad (37)$$

where $V^*(i,j)$ is the value function in each step, a is the maximum, and b is the minimum of $V^*(i,j)$, respectively. In principle, the inputs of the fuzzy inference engine are a neighbor of the robot's position. In the classical approach, only the closest robot neighbors are selected as optimal choices, while in the fuzzy Markov decision processes, the neighboring radius extends. Table 3 summarizes the square of the Euclidean distance between each cell and robot's position. This is amended by Figure 13 in which four neighborhoods with different radii are represented.

Table 3. Square of distance between each cell and robot's position.

	1	**2**	**3**	**4**	**5**
1	29	26	25	26	29
2	20	17	14	17	20
3	14	10	9	10	14
4	8	5	4	5	8
5	5	2	1	2	5
-	-	-	Robot	-	-

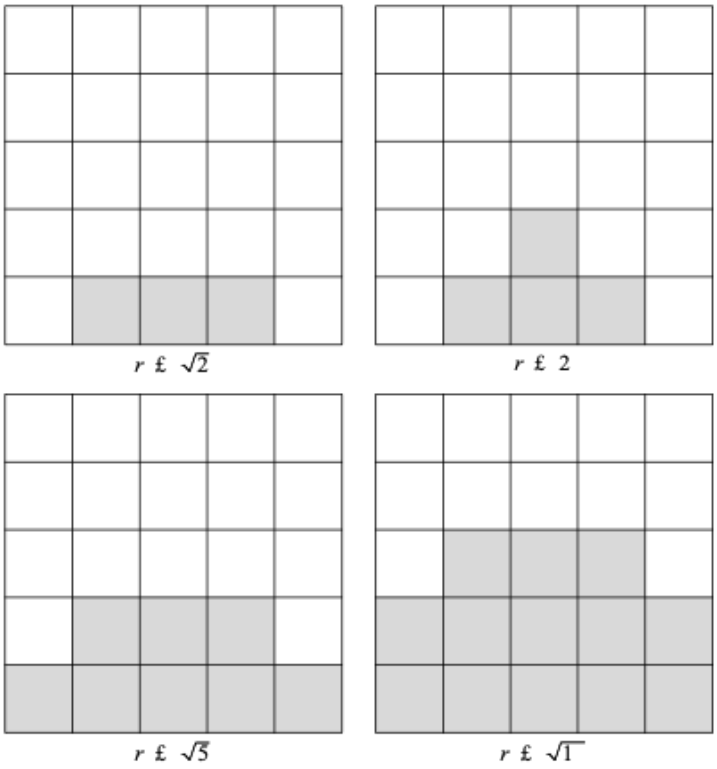

Figure 13. Four different neighborhoods of the robot in its frontal view.

Here, the neighborhood radius is assumed to be $\sqrt{10}$ and the inputs of the fuzzy inference engine are produced by Table 4.

Table 4. Inputs of the fuzzy inference method with neighborhood radius of $\sqrt{10}$ for a robot with 25 cells of view.

	1	**2**	**3**	**4**	**5**
1	-	-	-	-	-
2	-	-	-	-	-
3	-	X_1	X_2	X_3	-
4	X_4	X_5	X_6	X_7	X_8
5	X_9	X_{10}	X_{11}	X_{12}	X_{13}
-	-	-	Robot	-	-

To continue, the new value function is fuzzified. As an option, the value function can be considered as the membership function. This may result in over-fuzzification of the path which is undesired. To overcome this liability, other fuzzification functions, such as exponential, may be used. For example, it is possible to calculate the membership function as below:

$$\mu(A_i) = x_i^n \quad (38)$$

where x_i is the probability of x in cell i and n is an integer. Choosing the right integer requires experience and depends on the environment and camera of the robot. Each cell is dominated by a rule. The rule calculates the angle ϕ, which determines the direction of the robot's motion. The right-hand angles are defined positive, the left-hands angles are considered negative, and the head-on direction coincides with zero.

If A_1=1, then ϕ is a very small positive angle.

1. If A_2 = 1, then ϕ is zero.
2. If A_3 = 1, then ϕ is a very small negative angle.
3. If A_4 = 1, then ϕ is a medium positive angle.
4. If A_5 = 1, then ϕ is a small positive angle.
5. If A_6 = 1, then ϕ is a zero angle.
6. If A_7 = 1, then ϕ is a small negative angle.
7. If A_8 = 1, then ϕ is a medium negative angle.
8. If A_9 = 1, then ϕ is a big positive angle.
9. If A_{10} = 1, then ϕ is a medium positive angle.
10. If A_{11} = 1, then ϕ is a zero angle.
11. If A_{12} = 1, then ϕ is a medium negative angle.
12. If A_{13} = 1, then ϕ is a big negative angle.

From different existing defuzzification methods, the weighted average is chosen and used as follows:

$$\phi = \frac{\sum \mu(A_i)\hat{\phi}_i}{\sum \mu(A_i)} \quad (39)$$

where $\mu(A_i)$ is the membership function and $\hat{\phi}_i$ is the direction of the robot's motion.

Result of Fuzzy Markov Decision Processes in the Absence of Danger Space

The result of implementing the fuzzy Markov decision processes on the Aldebaran humanoid robot–Nao H25 is shown in Figure 14. As can be seen in this figure and the video associated with this figure, the robot successfully passes through obstacles and reaches the target.

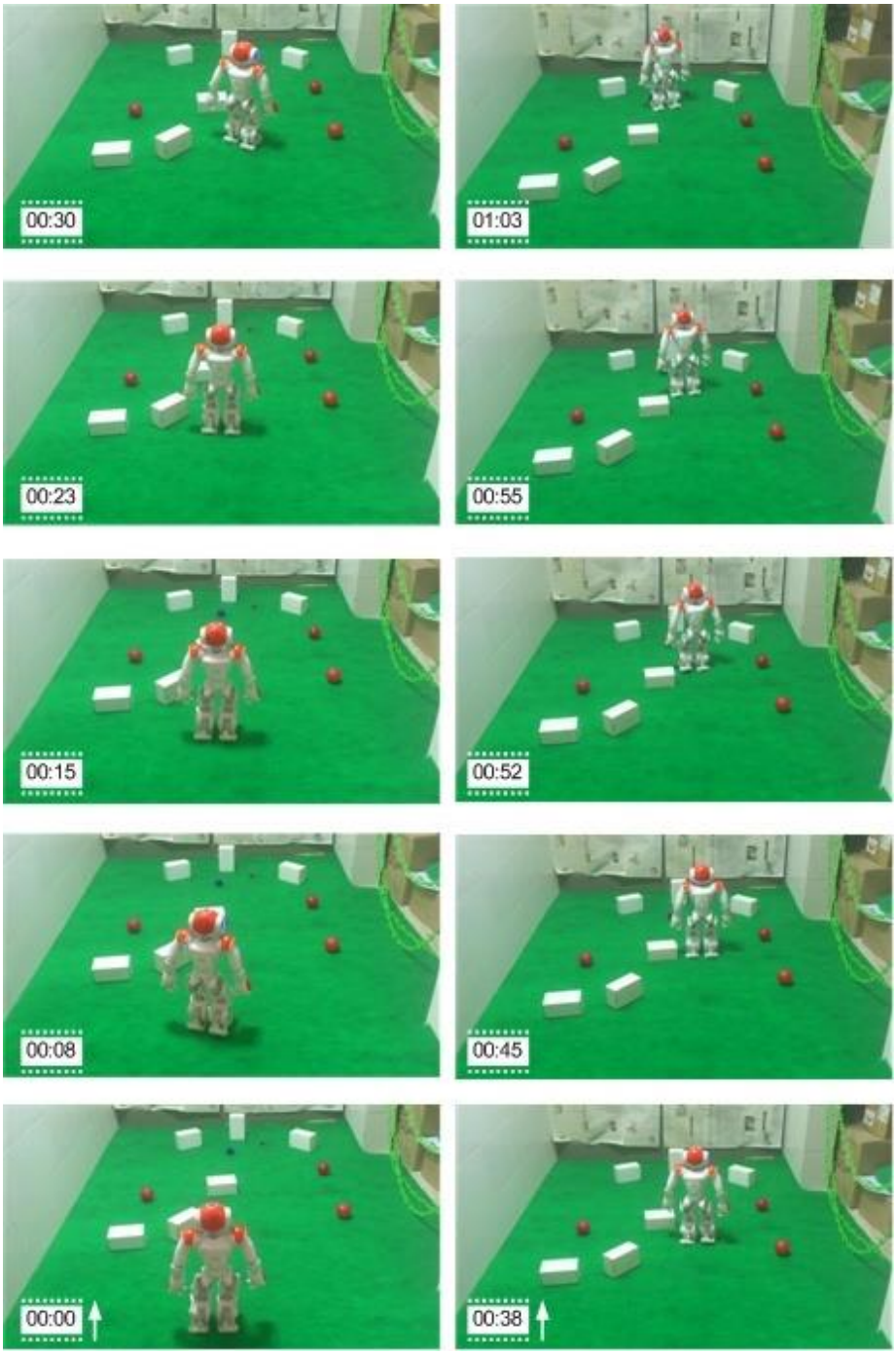

Figure 14. The path traversed by the Nao humanoid robot using the fuzzy Markov decision processes (see the video of the robot's movement).

4. Path Planning in the Presence of Danger Space

Generally, a robot encounters three types of environments: obstacle-free environment, obstacle, and target. While the robot may occasionally encounter another type of environment.

For example, in a wooded area, trees may be considered as obstacles, while mud pits are not blocking the robot's movement and the robot can pass through it. Meanwhile, it is not wise to choose a muddy path in the presence of the dry ground. In this case, mud pits should not be

regarded as the free spaces (as dry ground) nor obstacles (as trees). To solve this problem, a new space called danger space is introduced and added to the three traditional environments. Danger space is extendible to other cases like the greasy floor in a factory and areas in sight of an enemy in a battleground.

4.1. Disadvantages of Reward Calculation by Linear Relations

In the previous section, Equation (36) was proposed to calculate rewards in cells without a danger area. Given the linearity of the equations, this equation can be extended to the danger space as follows:

$$R(i,j) = P_{goal}(i,j) + w_1 P_{freespace}(i,j) + w_2 P_{danger}(i,j) + w_3 P_{obstacle}(i,j) \quad (40)$$

Although this equation appears to be effective, its linear properties may cause trouble. For example, while the coefficients of danger space and free space are close to each other, it may be preferable to cross the danger space to pass through the free space, which is definitely not desirable. Also, if the coefficients for the danger space and the space containing the obstacle are not different, when the passage of the danger space is the only option available, the robot may choose to traverse the obstacle and thus collide the obstacle. Therefore, an intelligent arrangement for determining rewards seems necessary.

4.2. Reward Calculation by the Fuzzy Inference System

Here, the Gaussian membership functions are used in fuzzification process. As can be seen in Figure 15, the space around each cell is a member of the fuzzy set including zero, small, medium, big, and very big.

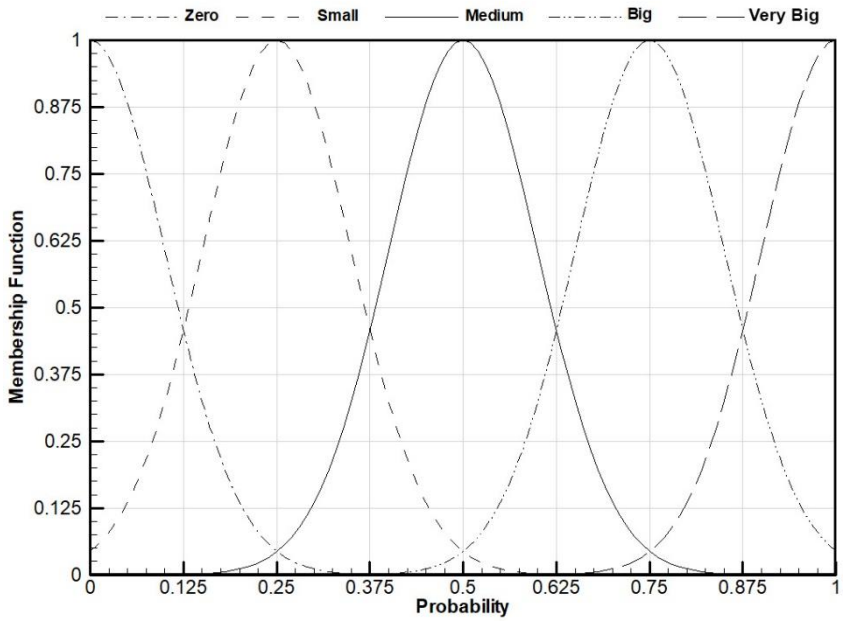

Figure 15. Fuzzy sets and membership functions of each space based on its probability.

The Takagi–Sugeno method is implemented to calculate rewards using the rules presented in Table 5. For example, as rule 1, if the probability of the existence of the obstacle is between 0.375 and 1, the probability of the existence of the danger is between 0 and 0.375, the probability of the existence of the free space is between 0 and 0.375, and the probability of the existence of the target is between 0 and 0.625, then the reward is considered as 0.1 Target.

Table 5. Fuzzy inference rule for reward.

Rule	Obstacle	Danger	Free	Target	Reward
1	Big	Zero	Zero	Small	0.1 Target
2	Medium			Medium	
3			Small	Small	
4		Small	Zero		
5	Small	Medium			
6		Zero	Medium		
7	Small	Small	Zero	Medium	0.2 Target
8		Zero	Small		
9	Small	Zero	Zero	Big	0.25 Target
10	Zero	Big	Zero	Small	0.333 Target
11	Zero	Medium	Small	Small	0.5 Target
12	Zero	Small	Medium	Small	0.667 Target
13	Zero	Medium	Zero	Medium	0.75 Target
14		Small	Small		
15		Zero	Big	Small	
16	Zero	Zero	Zero	Very Big	Target
17		Small		Big	
18		Zero	Medium	Medium	
19			Small	Big	
20	Small	Big	Zero	Zero	0.4 Obstacle
21		Medium	Small		
22		Small	Medium		
23	Medium	Small	Small	Zero	0.6 Obstacle
24	Big	Zero	Small	Zero	0.8 Obstacle
25	Medium		Medium		
26	Very Big	Zero	Zero	Zero	Obstacle
27	Big	Small			
28	Medium	Medium			
29	Zero	Small	Big	Zero	0.25 Danger
30	Zero	Medium	Medium	Zero	0.5 Danger

31	Zero	Very Big	Zero		
32	Small	Zero	Big	Zero	Danger
33	Zero	Big	Small		
34	Zero	Zero	Very Big	Zero	Free

Here, the weighted average method is used for defuzzification of reward as:

$$Reward = \frac{\sum \mu_i \hat{R}_i}{\sum \mu_i} \quad (41)$$

in which "Reward" is the defuzzification of reward, μ_i is the membership function, and $\hat{R}_i$ is the reward of each fuzzy rule.

4.3. Schematic Structure of Fuzzy Markov Decision Processes

After producing the final image, the reward associated with each part of the image is calculated. After that, Markov decision processes serve as an input for the fuzzy inference system. The output of the fuzzy inference system provides the robot with the necessary information for deciding on the direction and its movement. The schematic structure of this process is illustrated in Figure 16.

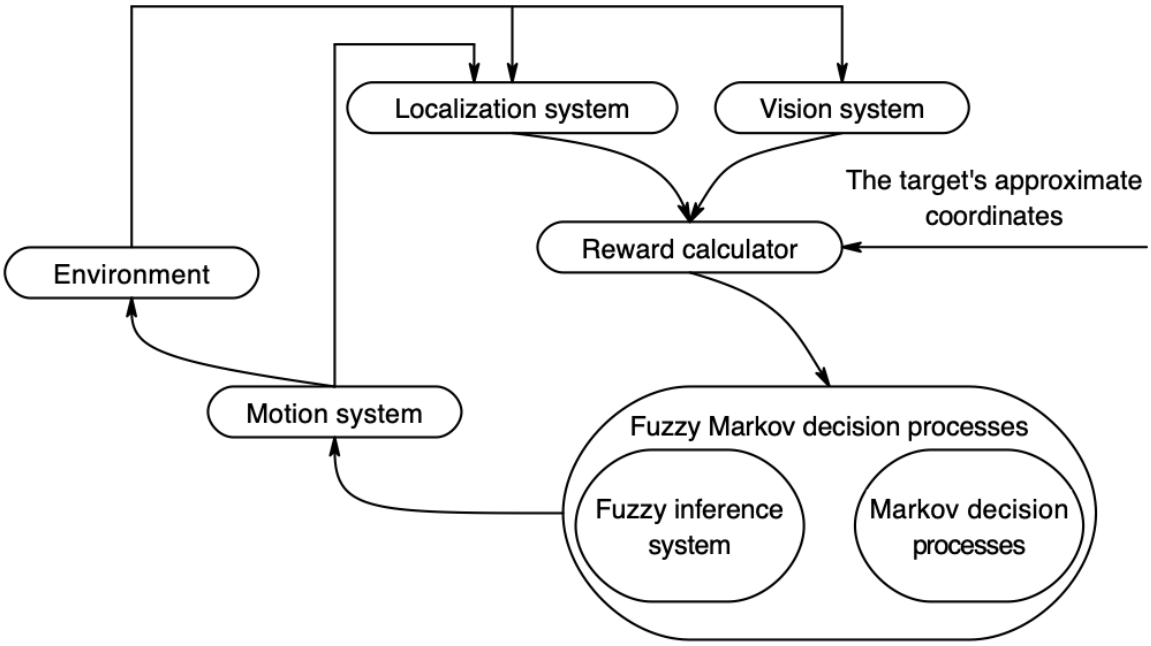

Figure 16. Schematic structure of fuzzy Markov decision processes.

4.4. Results of Fuzzy Markov Decision Processes in the Presence of Danger Space

The results of implementing the fuzzy Markov decision processes on the Nao humanoid robot are shown in Figures 17 and 18. As shown in Figure 17 and the video associated with this figure, the robot at the beginning of the path, considering that it encounters obstacles and danger space, selects the passing of the danger space as the only available option. After that, the robot continues to move toward the target. As the robot approaches the next danger space and given the availability of free space, it tries to avoid the danger space and move toward the target. Additionally, the robot tries to select the optimal path to reach the target. At this time, the robot takes a step to the right and, assuming that it has been able to terminate the danger space, turns to the left and goes to the

target. This causes the robot to touch the danger space when rotated to move toward the target. This problem can be solved by expanding the danger space. In Figure 18, the robot bypasses the danger space and successfully reaches the target.

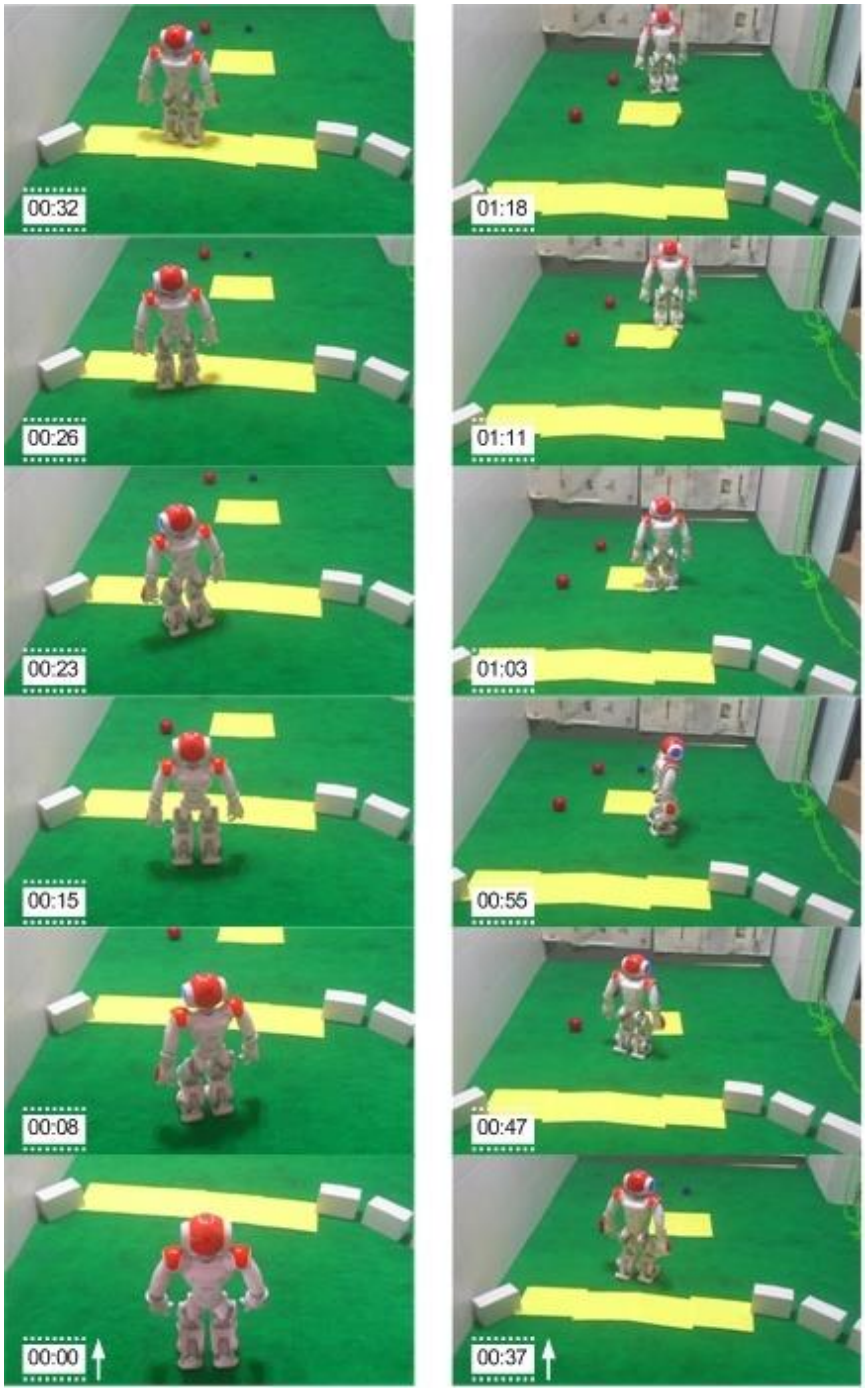

Figure 17. The path traversed by the Nao humanoid robot in presence of danger space using the fuzzy Markov decision processes in the first sample environment (see the video of the robot's movement).

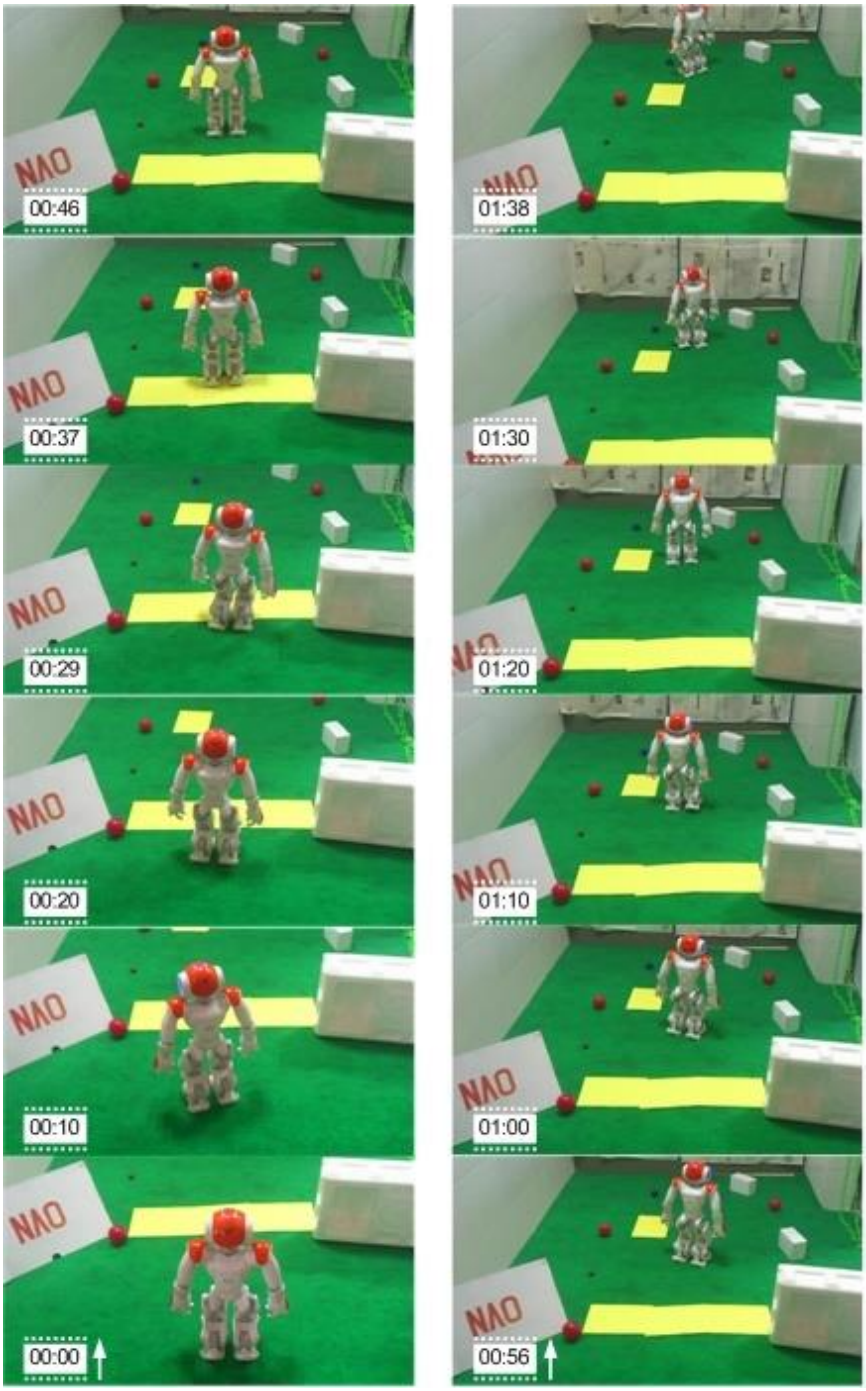

Figure 18. The path traversed by the Nao humanoid robot in presence of danger space using the fuzzy Markov decision processes in the second sample environment (see the video of the robot's movement).

As the final discussion, the use of the Markov decision processes leads to faster performance compared to the other proposed methods. In addition, the use of the fuzzy inference system leads to a smoother optimal path than previous ones. Moreover, the fuzzy Markov decision processes makes it possible to design a path without the need for accurate information on the shape, position, and orientation of the obstacles, as well as the need for having enormous volumes of memory to store data collected from two-dimensional and three-dimensional maps.

5. Conclusions

The present study addressed the path planning of the humanoid robot in the complex and unknown environments regarding the absence and presence of the danger space. A robot encounters three types of environments: obstacle-free environment, obstacle, and target. However, some spaces cannot be included in these three categories. In this regard, danger space was defined as specific space (like mud pits in a wooded area and greasy floor in a factory) which the robot is only permitted to be cross when no other options are available. Actually, the danger spaces are the potentially risky areas of the map. In the free-danger environment, synthetic potential field method was described and the governing equations were derived. To modify the inefficiency of this method in close proximity to the obstacles, a rectifier was introduced. The Linguistic method and Markov decision processes were other methods that are used for path planning in free-danger environments. A hybrid of Markov decision processes and fuzzy inference system was implemented to find an optimal and safe path from the start point to the target point in both environments, in the presence and absence of the danger space. This method improved the performance of the traditional Markov decision processes. Additionally, in order to real-time solving the Bellman equation, the value iteration was used. This method has been developed and successfully tested on an experimental humanoid robot (Nao H25 V4). As a future suggestion, the hybrid path planning algorithms using adaptive fuzzy membership functions can be implemented to create an optimal and safe path.

Author Contributions: Conceptualization, H.J.; Investigation, H.J., M.J. and N.N.S.; Methodology, M.J.; Resources, Viet-Thanh Pham; Software, M.J., N.N.S. and V.-T.P.; Supervision, X.Q.N.; Validation, V.V.H.; Writing—original draft, H.J., V.V.H.; Writing—review and editing, X.Q.N.

Funding: Please add: "This research received no external funding" or "This research was funded by [name of funder] grant number [xxx]" and "The APC was funded by [XXX]".

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lumelsky, V.J.; Skewis, T. Incorporating range sensing in the robot navigation function. *IEEE Trans. Syst. Man Cybern. Part A Syst. Humans* **1990**, *20*, 1058–1069.
2. Taylor, K.; LaValle, S.M. I-Bug: An intensity-based bug algorithm. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 2009; pp. 3981–3986.
3. Buniyamin, N.; Ngah, W.W.; Sariff, N.; Mohamad, Z. A simple local path planning algorithm for autonomous mobile robots. *Int. j. syst. Appl., Eng. Dev.* **2011**, *5*, 151–159.
4. Guruprasad, K.R. EgressBug: a real time path planning algorithm for a mobile robot in an unknown environment. In *Advanced Computing, Networking and Security*; Springer: Berlin, Germany, 2011; pp. 228–236.
5. Xu, Q.-L.; Tang, G.-Y. Vectorization path planning for autonomous mobile agent in unknown environment. *Neural Comput. Appl.* **2013**, *23*, 2129–2135.
6. Hernandez, E.; Carreras, M.; Ridao, P. A comparison of homotopic path planning algorithms for robotic applications. *Rob. Autom. Syst.* **2015**, *64*, 44–58.
7. González, D.; Pérez, J.; Milanés, V.; Nashashibi, F. A Review of Motion Planning Techniques for Automated Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 1135–1145.
8. Ravankar, A.; Ravankar, A.; Kobayashi, Y.; Hoshino, Y.; Peng, C.-C. Path smoothing techniques in robot navigation: State-of-the-art, current and future challenges. *Sensors* **2018**, *18*, 3170.
9. Sarkar, B.; Guchhait, R.; Sarkar, M.; Pareek, S.; Kim, N. Impact of safety factors and setup time reduction in a two-echelon supply chain management. *Rob. Comput. Integr. Manuf.* **2019**, *55*, 250–258.
10. Song, R.; Liu, Y.; Bucknall, R. Smoothed A* algorithm for practical unmanned surface vehicle path planning. *Appl. Ocean Res.* **2019**, *83*, 9–20.

11. Fu, B.; Chen, L.; Zhou, Y.; Zheng, D.; Wei, Z.; Dai, J.; Pan, H. An improved A* algorithm for the industrial robot path planning with high success rate and short length. *Rob. Autom. Syst.* **2018**, *106*, 26–37.
12. Foux, G.; Heymann, M.; Bruckstein, A. Two-dimensional robot navigation among unknown stationary polygonal obstacles. *IEEE Trans. Rob. Autom.* **1993**, *9*, 96–102.
13. Choset, H.M.; Hutchinson, S.; Lynch, K.M.; Kantor, G.; Burgard, W.; Kavraki, L.E.; Thrun, S. *Principles of Robot Motion: Theory, Algorithms, and Implementation*; MIT Press: Cambridge, MA, USA, 2005.
14. Jarris, R.A. Collission-free trajectory planning using distance transforms. *Mech. Eng. Trans. IE Aust.* **1985**, *10*, 187.
15. Sudhakara, P.; Ganapathy, V.; Priyadharshini, B.; Sundaran, K. Obstacle Avoidance and Navigation Planning of a Wheeled Mobile Robot using Amended Artificial Potential Field Method. *Procedia Comput. Sci.* **2018**, *133*, 998–1004.
16. Moreau, J.; Melchior, P.; Victor, S.; Aioun, F.; Guillemard, F. Path planning with fractional potential fields for autonomous vehicles. *IFAC-PapersOnLine* **2017**, *50*, 14533–14538.
17. Zhou, Z.; Wang, J.; Zhu, Z.; Yang, D.; Wu, J. Tangent navigated robot path planning strategy using particle swarm optimized artificial potential field. *Optik-Int. J. Light Electron. Opt.* **2018**, *158*, 639–651.
18. Matoui, F.; Boussaid, B.; Metoui, B.; Frej, G.B.; Abdelkrim, M.N. Path planning of a group of robots with potential field approach: Decentralized architecture. *IFAC-PapersOnLine* **2017**, *50*, 11473–11478.
19. Bayat, F.; Najafinia, S.; Aliyari, M. Mobile robots path planning: Electrostatic potential field approach. *Expert Syst. Appl.* **2018**, *100*, 68–78.
20. Larsen, L.; Kim, J.; Kupke, M.; Schuster, A. Automatic Path Planning of Industrial Robots Comparing Sampling-Based and Computational Intelligence Methods. *Procedia Manuf.* **2017**, *11*, 241–248.
21. Abdelwahed, M.F.; Saleh, M.; Mohamed, A.E. Speeding up single-query sampling-based algorithms using case-based reasoning. *Expert Syst. Appl.* **2018**, *114*, 524–531.
22. Perez-Lozano, T. Spatial planning: A configuration space approach. In *Autonomous Robot Vehicles;* Springer: New York, NY, USA, 1990.
23. Schwartz, J.T.; Yap, C.-K. *Algorithmic and Geometric Aspects of Robotics (Routledge Revivals)*; Routledge: British, UK, 2016.
24. Schwartz, J.T.; Sharir, M. On the "piano movers" problem. II. General techniques for computing topological properties of real algebraic manifolds. *Adv. Appl. Math.* **1983**, *4*, 298–351.
25. Precup, R.-E.; Hellendoorn, H. A survey on industrial applications of fuzzy control. *Comput. Ind.* **2011**, *62*, 213–226.
26. Jafarzadeh, M.; Gans, N.; Tadesse, Y. Control of TCP muscles using Takagi–Sugeno–Kang fuzzy inference system. *Mechatronics* **2018**, *53*, 124–139.
27. Rajagopal, K.; Jahanshahi, H.; Varan, M.; Bayır, I.; Pham, V.-T.; Jafari, S.; Karthikeyan, A. A hyperchaotic memristor oscillator with fuzzy based chaos control and LQR based chaos synchronization. *AEU Int. J. Electron. Commun.* **2018**, *94*, 55–68.
28. Jahanshahi, H.; Rajagopal, K.; Akgul, A.; Sari, N.N.; Namazi, H.; Jafari, S. Complete analysis and engineering applications of a megastable nonlinear oscillator. *Int. J. Non Linear Mech.* **2018**, *107*, 126–136.
29. Mahmoodabadi, M.J.; Jahanshahi, H. Multi-objective optimized fuzzy-PID controllers for fourth order nonlinear systems. *Eng. Sci. Technol., Int. J.* **2016**, *19*, 1084–1098.
30. Zavlangas, P.G.; Tzafestas, S.G.; Althoefer, K. Fuzzy obstacle avoidance and navigation for omnidirectional mobile robots. In Proceedings of the ESIT'2000, Aachen, Germany, 2000; pp. 375–382.
31. Al Yahmedi, A.S.; Fatmi, M.A. Fuzzy logic based navigation of mobile robots. In *Recent Advances in Mobile Robotics*; IntechOpen: London, United Kingdom, 2011.

32. Iancu, I.; Colhon, M.; Dupac, M. A Takagi-Sugeno type controller for mobile robot navigation. In Proceedings of the 4th WSEAS international conference on computational intelligence, man-machine systems and cybernetics, Miami, FL, USA, 17–19 November, 2005; pp. 29–34.
33. Michel, P.; Chestnutt, J.; Kuffner, J.; Kanade, T. Vision-guided humanoid footstep planning for dynamic environments. In Proceedings of the 5th IEEE-RAS International Conference on Humanoid Robots, Tsukuba, Japan, 5 December 2005; pp. 13–18.
34. Nakhaei, A.; Lamiraux, F. Motion planning for humanoid robots in environments modeled by vision. In Proceedings of the 8th IEEE-RAS International Conference on Humanoid Robots, Daejeon, South Korea, 1–3 December 2008; pp. 197–204.
35. Sabe, K.; Fukuchi, M.; Gutmann, J.S.; Ohashi, T.; Kawamoto, K.; Yoshigahara, T. Obstacle avoidance and path planning for humanoid robots using stereo vision. In Proceedings of the IEEE International Conference on Robotics and Automation, New Orleans, LA, USA, 26 April–1 May 2004; pp. 592–597.
36. Michel, P.; Chestnutt, J.; Kagami, S.; Nishiwaki, K.; Kuffner, J.; Kanade, T. Online environment reconstruction for biped navigation. In Proceedings of the IEEE International Conference on Robotics and Automation, Orlando, FL, USA, 15–19 May 2006; pp. 3089–3094.
37. Chestnutt, J.; Kuffner, J.; Nishiwaki, K.; Kagami, S. Planning biped navigation strategies in complex environments. In Proceedings of the IEEE International Conference on Humanoid Robotics, Taipei, Taiwan, 14–19 September 2003.
38. Okada, K.; Inaba, M.; Inoue, H. Walking navigation system of humanoid robot using stereo vision based floor recognition and path planning with multi-layered body image. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas, NV, USA, 27–31 October 2003; pp. 2155–2160.
39. Zhang, Y.; Gong, D.-W.; Zhang, J.-H. Robot path planning in uncertain environment using multi-objective particle swarm optimization. *Neurocomputing* **2013**, *103*, 172–185.
40. Purcaru, C.; Precup, R.-E.; Iercan, D.; Fedorovici, L.-O.; David, R.-C. Hybrid PSO-GSA robot path planning algorithm in static environments with danger zones. In Proceedings of the 17th International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania, 11–13 October 2013; pp. 434–439.
41. Zhang, H.-M.; Li, M.-L.; Yang, L. Safe Path Planning of Mobile Robot Based on Improved A* Algorithm in Complex Terrains. *Algorithms* **2018**, *11*, 44.
42. Cheng, D.K. *Field and Wave Electromagnetics*; Addison-Wesley: Boston, USA, 1989.
43. Fakoor, M.; Kosari, A.; Jafarzadeh, M. Revision on fuzzy artificial potential field for humanoid robot path planning in unknown environment. *Int. J. Adv. Mechatron. Syst.* **2015**, *6*, 174–183.
44. NAO-Construction. Availabe online: http://doc.aldebaran.com/2-1/family/robots/dimensions_robot.html (accessed on 17 August 2018)
45. Bellman, R. A Markovian decision process. *J. Math. Mechanics* **1957**, 679–684.
46. Kolobov, A. Planning with Markov decision processes: An AI perspective. *Synth. Lect. Artif. Intell. Mach. Learn.* **2012**, *6*, 1–210.
47. Fakoor, M.; Kosari, A.; Jafarzadeh, M. Humanoid robot path planning with fuzzy Markov decision processes. *J. appl. Res. Technol.* **2016**, *14*, 300–310.
48. Hu, Q.; Yue, W. *Markov Decision Processes withTheir Applications*; Springer Science & Business Media: Berlin, Germany, 2007.
49. Puterman, M.L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*; John Wiley & Sons: Hoboken, USA, 2014.
50. Sheskin, T.J. *Markov Chains and Decision Processes for Engineers and Managers*; CRC Press: Boca Raton, USA, 2016.