International Journal of Applied Mathematics

Volume 27 No. 1 2014, 73-88
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)
doi: http://dx.doi.org/10.12732/ijam.v27i1.8

MATHEMATICAL MORPHOLOGY AND ARTIFICIAL INTELLIGENCE APPLIED TO HELP GOLF-BALLS COLLECTING IN DRIVING RANGES

Maximilian Jaderson de Melo[1], Maurıcio Araújo Dias[2 §],
Almir Olivette Artero[3], Erivaldo Antonio da Silva[4]

[1,2,3]Department of Mathematics and Computing
Department of Cartography
School of Science and Technology
São Paulo State University (Univ. Estadual Paulista - UNESP)
Roberto Simonsen street, 305
Presidente Prudente, SP, 19060-900, BRAZIL

Abstract: Manual golf-balls collecting in indoor driving ranges is a tedious task. The objective of this paper is to propose a method that contributes to automate the golf-balls collecting in indoor driving ranges. This method applies Mathematical Morphology and Artificial Intelligence to help the balls collecting. The proposal is represented by a program based on Mathematical Morphology applied to balls detection and Genetic Algorithm applied to solve the Travelling Salesman Problem to conduct a virtual robot for balls collecting in a virtualized intelligent indoor driving range. Images of balls scattered on the ground were processed to obtain the results. The results show the relevance of our project as a method to help balls collecting in indoor driving ranges. Our proposal contributes to the implementation of a low computational-cost method for automatic organization of driving ranges and to low computational-cost intelligent environments.

AMS Subject Classification: 03E99, 68U10, 93C85, 68T20
Key Words: automation, mathematical morphology, artificial intelligence, golf-balls collecting, driving range, genetic algorithm

Received: January 17, 2014

§Correspondence author

1. Introduction

Golf-balls collecting in indoor driving ranges for golf swing is a repetitive and boring task. Golf driving ranges are usually small, flat and covered, unlike golf courses which are usually huge and have several obstacles as slight undulations in relief, sandy ground, lakes or rivers and vegetation. This fact facilitates its organization, but the collecting of the scattered golf-balls is often a manual and hard task that requires time and also interrupts the time that golfers have to practice. Alternatively, some projects for automatic golf-balls collecting have been developed in the past years.

One of these projects developed a robot to pick up the balls from the ground and put them into goals [1], [5]. It uses an autonomous robot equipped with a infra-red sensors set to follow a grid of white lines drawn on the lawn. The robot can only collect balls that are near the white lines. Unfortunately, to draw and maintain those lines all over the range would be more costly and boring than simply to collect the golf-balls manually.

Other project presents a golf-balls collecting automated tractor vehicle [4]. The vehicle is equipped with a laser scanner, omnidirectional camera, orientation sensor (compass) and Global Positioning System (GPS). The vehicle avoids obstacles, collects golf-balls, but initial path must be provided manually. Moreover it requires a substantially expensive hardware, which would costs at least a few thousands of dollars.

A third project addresses issues on localization of targets using RGB video stream [9]. The authors acquire one RGB image per second from a web cam. Once each image is acquired, the authors apply two band gaps threshold in the image due to detect objects. Once objects are successfully detected, boundary is extracted and labeled. Then, the authors estimate targets distances. In perspective view, objects sizes are smaller as much as their distances grow from observer. Given that, the authors use a function to estimate objects distances based on objects scales. Unfortunately, the processing of color images is slower than the processing of binary images. Moreover this project does not perform path planning.

In other project, a global vision-based golf-balls collecting robot is presented [16]. The system localizes golf-balls with a global camera that delivers real-time images to a server. To accomplish this, the system applies image processing techniques to retrieve balls positions. The system make the path planning identifying dense balls regions and then tracing a graph based on Travelling Salesman Problem (TSP) [14], for which each region represents a city. The system performs the balls collecting only with a spiral scanning strategy to

each effective region on the field. Unfortunately, this strategy leaves the balls that are out of the spiral regions on the field.

To solve the aforementioned problems, like manual and boring golf-balls collecting, drawn guide-lines on ground, high cost-complex vehicles and robots, computational-expensive processing of color images and absence of path planning, this paper proposes a method that contributes to automate the golf-balls collecting in indoor driving ranges. This method applies Mathematical Morphology and Artificial Intelligence to help balls collecting. The proposal is represented by a program based on Mathematical Morphology applied to balls detection and Genetic Algorithm applied to solve the Travelling Salesman Problem to conduct a virtual robot for balls collecting in a virtualized intelligent indoor driving range.

This method solves all problems cited above. For example: the method eliminates the manual and boring balls collecting without any kind of ground marking; the image processing is binary, reducing the computational-cost; the method determines the optimal collection path to the robot; and all hard processing is server-sided, which reduces the complexity and cost of the robot.

To better explain our proposal, this paper is organized as follows. Section 2 describes theoretical fundamentals. Section 3 describes the proposed method. Section 4 presents the results. Section 5 presents discussions and conclusions.

2. Theoretical Fundamentals

As already commented, our method is based on Mathematical Morphology applied to balls detection and Genetic Algorithm applied to solve the Travelling Salesman Problem to conduct a virtual robot for balls collecting in an Intelligent Environment. In the next subsections, we present very brief theoretical fundamentals about: Mathematical Morphology, Travelling Salesman Problem, Genetic Algorithm and Intelligent Environment.

2.1. Mathematical Morphology

The Mathematical Morphology (MM) focuses on the geometry and structure of the image [3], [8]. The MM basic idea is to apply a structuring element (SE) to a set of pixels (the image) and analyze the manner or amount that the SE fits or not in the image. A SE is basically a smaller image in which the pixels are disposed in some basic shape into a rectangular matrix. The shape usually is: cross, disk, square, rectangle, line or diamond. If we apply a SE in

an image, the locations where the SE fits represent the interesting information. This information depends on both size and shape of the SE. Even when using machine learning, choose the structuring element depends specifically on the type of information expected to be derived. For our method, we apply a SE B to perform some MM operations, as erosion, dilation and opening, to an image A.

2.1.1. Erosion

Erosion is the most elemental operation of MM. Given a set A and a SE B, the erosion of an image A by a SE B can be described as:

$$A \ominus B = \{x : B_x \subset A\}, \tag{1}$$

for which $\subset$ represents a subset relation. In other words, the erosion of A by the SE B is the set of every point x, where B translated by x, is contained in A.

2.1.2. Dilation

The dilation is other elemental operation of MM. The equation that represents the dilation of a set A by a SE B is:

$$A \oplus B = \{x : (\breve{B})_x \cap A \neq \emptyset\}. \tag{2}$$

The dilation can be obtained with erosion by set completion. Thus, given a set A and a SE B, the dilation $A \oplus B$ is also described as:

$$A \oplus B = (A^c \ominus \breve{B})^c, \tag{3}$$

where A^c represents the complement of the image A. If B contains the origin, dilating A by B results in a expansion of A. To dilate A by B, B is rotated around the origin to obtain $\breve{B}$. Dilation is the erosion of A^c by $\breve{B}$, followed by the complement of this erosion. Given a SE, dilation fills small protrusions and holes into an image, unlike the erosion that removes small components.

2.1.3. Opening

Given the set A as an image and B as the SE, the opening is denoted by:

$$A \circ B = (A \ominus B) \oplus B. \tag{4}$$

Another possible opening equation is:

$$A \circ B = \bigcup \{B_x : B_x \subset A\}. \tag{5}$$

2.2. Travelling Salesman Problem

The Travelling Salesman Problem (TSP) is a classical NP-Hard combinatorial optimization problem [14]. There is not a recognized exact algorithm able to solve this problem in an acceptable amount of time when the number of vertexes is substantially big. The TSP tries to determine the best path, given a set of cities and their distances to each other. Each city must be visited once and the salesman must return to the origin lastly.

Let us consider: a graph $G = (NE)$, where $N = \{1, \ldots, N\}$ is the set of nodes and $E = \{1, \ldots, M\}$ is the set of inter-city links; c_{ij} is the cost of each inter-city link to the vertexes i and j, $c_{ij} = c_{ji}$ and $c_{ii} = 0$. The problem consists to discover the smallest Hamiltonian circuit [14]. The circuit size is defined by the sum of each inter-city link contained in the circuit. The ideal path must minimize the whole distance travelled.

The size of the space of solutions increases exponentially as the number of cities grows. The number of possible circuits is given by:

$$\frac{(n-1)!}{2} \approx \frac{1}{2}\sqrt{2\pi(n-1)}\left(\frac{n-1}{e}\right)^{n-1}. \tag{6}$$

TSP can be modeled as an undirected weighted graph, for which each vertex represents a city and the inter-city link between two vertexes represents two interconnected cities. The weight in an inter-city link represents the distance between two cities. Figure 1 shows a graph representation for four cities.

2.3. Genetic Algorithms

A genetic algorithm (GA) is an algorithm that mimics evolution and is classified as an heuristic algorithm [15]. The GA are inspired by the Darwinism, for which the chances of an individual are as large as its adjustment to its environment. Usually, genetic algorithms present better results for problems of combinatorial nature [11], which justifies the use of GA to discover approximate solutions to optimization and to search problems. The simplest representation of a GA is an array of bits. An array of bits is named **chromosome** and each bit is called **gene**. The **population** consists of a set of chromosomes. The initial population can be randomly generated, but it is recommended to use initial solutions

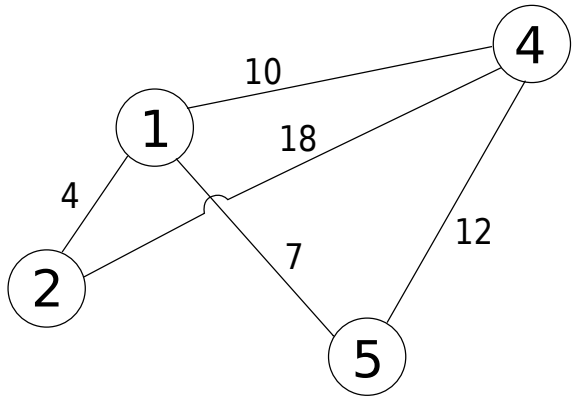

Figure 1: TSP graph representation

that involve some kind of heuristics, or ensuring that the population is evenly distributed among the space of solutions for better results. The evaluation of chromosomes is done by a fitness function, that measures the individual ability to survive and reproduce. The fitness function is a particular type of objective function and must be maximized or minimized depending on the problem.

2.4. Intelligent Environment

According to robotics, an environment can be called an intelligent environment (IE) when all its resources are perfectly used to enhance the activities of an agent, and at the same time are unnoticeable by the agent [2]. In other words, in an IE, most of intelligence is server sided by a computer, which communicates with the robot often by radio.

3. Materials and Methods

We used a computer running 32-bits Ubuntu Linux, version 12.04. We implemented the proposed method as a program. This program was coded in C++, using the cross-platform development framework Qt Creator, version 2.7.2. All operations of image processing was implemented using the Open Source Computer Vision Library (OpenCV). Our program works as represented by Figure 2.

In the first step, our program acquires an image from a global camera placed above the field, as represented by Figure 3. An example of an image acquired

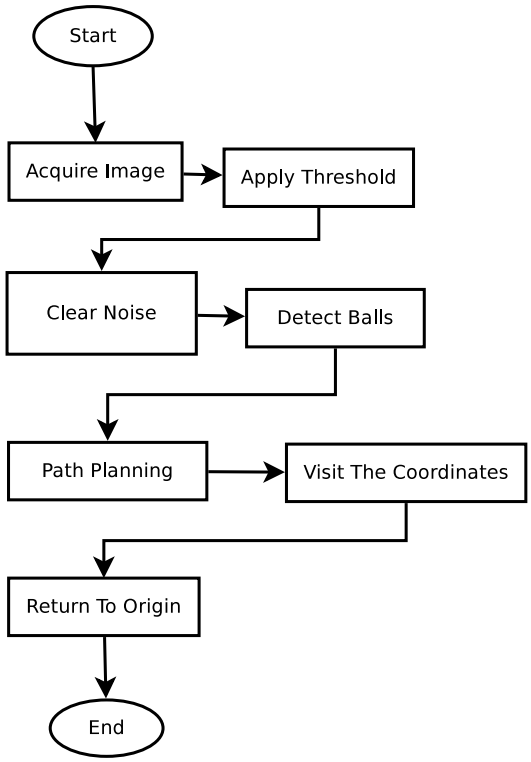

Figure 2: Flowchart of our program

by the global camera is showed by Figure 4. Next, our program converts the image from RGB to a grayscale image, with the OpenCV function: cvtColor(InputArray src, OutputArray dst, int code, int dstCn=0), in which the parameters are:

- **src**: is an 8-bit unsigned or 16-bit unsigned input image;
- **dst**: is the output image, with the same depth as src;
- **code**: is the code for conversion in the color space. We used CV_BGR2GRAY. It means that the src is a BGR and will be transformed into a grayscale image;
- **dstCn**: is the number of channels of dst; when 0, the number of channels will be obtained automatically from src and code.

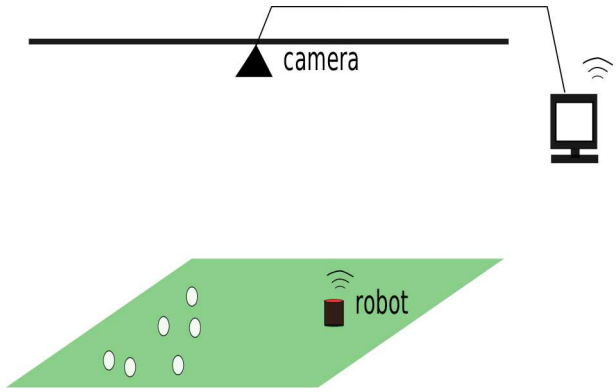

Figure 3: Global camera

Then our program applies a threshold in the grayscale image, using the level 216. This level was chosen because this value allowed detecting the highest number of objects and the smallest amount of noise for the images used in the experiments of this work. This operation results in a binary image, in which all pixels with value bigger than the level will be changed to one (1), zero (0) otherwise.

After this processing, only golf-balls and some salt noise remain in the image as sets of white pixels. The noise results from the conversions between different types of images. This noise is removed in the step Clear Noise, by applying a MM operation.

Figure 4: Acquired image example

Specifically, in step Clear Noise, our program applies an image opening with a cross shaped SE with dimensions of 4X4 pixels, with the OpenCV function morphologyEx. We provide the following paramenters to this function:

- **src**: the input image;
- **dst**: the output image;
- **op**: the operation (in our case, the Opening);
- **element**: the structuring element;

These parameters allow removing the noise, maintaining the original size of the golf-balls present in the image. Figure 5 shows a close view of the golf-balls present in the acquired image, just after the steps Apply Threshold and Clear Noise. The image is color-inverted to improve visualization.

In the step Detect Balls, our program applies an erosion with a cross shaped SE with dimensions of 5X5 pixels. This operation reduces the number of white pixels of the golf-balls that need to be processed in the next phase of this same step.

Then, our program finds the locations of the golf-balls, first by using a function to find contours, next by finding the center of mass of each ball. We used the OpenCV function findContours(InputOutputArray image, OutputArrayOfArrays contours, OutputArray hierarchy, int mode, int method, Point offset=Point()). The parameters are:

- image: it is the source image.

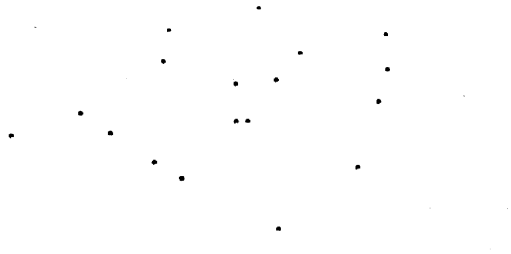

Figure 5: Close view of golf-balls in the acquired image after steps Apply Threshold and Clear Noise. The image is color-inverted

- contours: It is the output vector containing the contours present in the image. These contours are stored as a vector of points.
- hierarchy: it is optional and stores information about image topology.
- mode: It is the contour finding mode. We used tree mode (CV_RETR_TR EE), that find all contours and creates a full hierarchy of nested contours.
- method: It is the contour approximation method. We used simple approximation
(CV_CHAIN_APPROX_SIMPLE);
- offset: it is optional, and represents the offset for which all contours are shifted.

Figure 6 shows a close view of the golf-balls present in the acquired image, just after the step Detect Balls. We have changed the original black background of the image to a white background only to improve visualization.

In the step Path Planning, our program makes the path planning considering the shortest path strategy. Our program implements TSP with the aid of a version of GA. In our GA implementation, we considered each ball as a city (a TSP vertex). Each vertex is enumerated ranging from zero to the total number of balls less one. After discovering all vertexes, they are connected until form an undirected graph.

We represented the genes of the GA with numeric representation. The size of the chromosome is dynamic, since its value depends on the number of balls found in the step Detect Balls. However, once it is defined, it cannot be changed.

In our implementation, there are some parameters that must be provided in advance, like number of chromosomes (population size) N, maximum number of

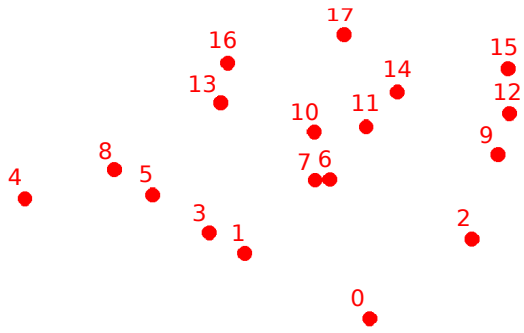

Figure 6: Close view of the golf-balls after the step Detect Balls (background changed to white to improve visualization)

generations MG, maximum number of generations without any improvement NUT, mutation rate MR, elitism rate ER. The parameters supplied was: $N = 100$, $MG = 3000$, $NUT = 1500$, $MR = 3\%$ and $ER = 15\%$. These values were empiric found and they were chosen because they allow achieving the best results.

To the initial population, our program creates a starting chromosome based on the nearest neighbor algorithm that grants at least a local maximum solution. With this initial chromosome, our program randomly changes genes of this chromosome generating a new chromosome. Then, our program adds this new chromosome in the population list. These actions repeat until the population reaches the population size N. Algorithm 1 represents this logic.

Algorithm 1 Initializing population

```
1: discoverOfInitialChromosomec
2: while chromosomes.size < N do
3:     nc ← c
4:     for all gene of nc do
5:         swap ← random
6:         if swap = true then
7:             nc ← swap(c, random, random)   ▷ we treated cases of repeated
   indices
8:         end if
9:     end for
10:    chromosomes.add(nc)
11: end while
```

Our program uses roulette as method of parents selection. This method assigns a bigger chance to hit an individual with better fitness, but does not guarantee that this individual will always be chosen. This feature is responsible for the diversity of solutions. Our program spins the roulette twice and the selected genes are combined. To combine two genes, our program uses the operator OX (Order Crossover). The operator OX ensures that will not occur any invalid combination.

There is also the mutation operator that is responsible for avoid that the GA stay much time stuck in a local optimal solution. The mutation operator implemented is reciprocal exchange.

For the fitness function, the objective function minimizes the path to collect the balls, departing from an origin point. The origin point is defined as the point $(0, 0)$ in the image. Since each ball is connected with all other balls, and each ball is represented by a pair of coordinates in the image, our program calculates the distance of a ball based on the sum of the Euclidean distances among a ball i and all other balls. Then, our program evaluates the fitness as the biggest distance minus the current distance:

$$\forall g, g \in Population,$$

$$g_i.dist = \sum_{j=0}^{k-1} \sqrt{(g_j.x - g_{j+1}.x)^2 + (g_j.y - g_{j+1}.y)^2} \tag{7}$$

$$\max Dist = \max(g.dist) \tag{8}$$

where g_i represents an element of the population of chromosomes, g_j represents a gene of the chromosome g_i, k represents the amount of genes of each chromosome and *maxDist* stores the biggest distance among all chromosomes,

$$g_i.fitness = \max Dist - g_i.dist\,. \tag{9}$$

We used elitism to ensure that the best chromosomes found will not be lost during any generation of the crossover. The elitism was implemented using a simple bubble sort function that considers the fitness of each chromosome as sort criteria. The sort criteria is activated always after computing the fitness of the chromosomes in each generation. After computing fitness and just before the population selection, the best individuals of the population (15%) are automatically sent to the next generation.

At the end of the step Path Planning, our program puts the most evolved gene that contains the approximated-shortest path in the first position of the population. Figure 7 shows a close view of the golf-balls present in the acquired

image, just after the step Path Planning. We have changed the original black background of the image to a white background only to improve visualization. The shortest path is not always obtained because of the randomly nature of GA.

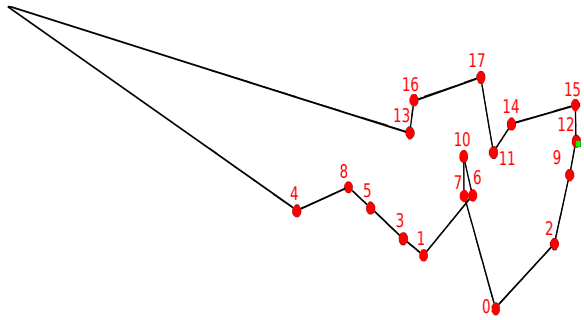

Figure 7: Path planning example (background changed to white to improve visualization)

In the step Visit the Coordinates, our program sends to a virtual robot the coordinates of the balls obtained in the prior step. These coordinates are in the chromosome, inside the genes. This step guides the robot to each ball. We basically iterate a loop for all pair of coordinates of the balls contained inside the genes in an orderly manner. For each ball, our program begins another loop, increasing or decreasing the current position of the robot at that moment, according to the difference between the coordinates of the robot and the coordinates of the next ball in the path. Figure 7 shows the virtual robot visiting the ball number 13. The robot is represented by a green square. Finally, when the virtual robot reaches the last ball, our program sends it back to the origin.

4. Results

We used an image database containing twenty six RGB images, each one with 2592 x 1944 pixels and size 2MB, in the experiments. Those images contain eighteen balls, in average, arbitrarily scattered in the range. The performance of our program to detect balls was validated with the following metric [13], [12]:

$$M_B = \frac{TP_B + TN}{TP_B + TN + FN_B + FP_B}, \tag{10}$$

where M_B indicates the success rate; TP_B is the number of true positives or the number of pixels of balls identified by the program, or the balls correctly identified; TN is the number of true negatives, or the number of pixels of the ground correctly identified; FN_B is the number of false negatives, or pixels of balls that were mistakenly identified as ground and FP_B is the number of false positives that is the number of pixels of the ground identified as pixels of balls by the program.

Based on this metric, our program achieved a success rate of 99% in balls detection. The collecting reached a rate of 100%, since all balls detected was virtually collected. The implemented GA does not guarantee that the best solution will be reached, but at least a local minimum will be obtained at the end of the execution.

Note that our program was successful both in detection, based on MM, and in the virtual collecting, based on TSP and GA.

5. Discussion

We presented a method that contributes to automate the golf-balls collecting in indoor driving ranges. This method applies Mathematical Morphology and Artificial Intelligence to help balls collecting. The proposal is represented by a program based on Mathematical Morphology applied to balls detection and Genetic Algorithm applied to solve the Travelling Salesman Problem to conduct a virtual robot for balls collecting in a virtualized intelligent indoor driving range.

This method overcomes problems related by others researchers in [1], [5], [4], [9] and [16], like manual and boring golf-balls collecting, drawn guide-lines on ground, high cost and complex vehicles and robots, computational-expensive processing of color images and absence of path planning, which were commented in the introduction of this paper. The results achieved by our program indicates balls detection success rate of 99%, comparable to [9], but without the need of any kind of color image processing. Moreover, our path planning finds always at least an acceptable solution to collect all detected balls.

Our method offers the following contributions: it avoids the manual and boring golf-balls collecting; it makes all the image processing in binary mode and uses images instead of video [9], which ensures a low computational-cost image processing; the complexity and cost of the robot are significantly low, since the robot needs only to know about movement tasks; the path planning is robust and able to determine a solution close to the optimal; the path planning

does not need any ground marking.

6. Conclusions

We conclude that our method successfully contributes to automate the golf-balls collecting in indoor driving ranges.

In future researches, we intend to perform tasks that address: the removal of the shadow effects in the range; the path planning upgrading by other heuristic like Ant Colony Optimization (ACO) [10], Tabu Search (TS) [7] or Scatter Search (SS) [6]; the use of threads to accomplish faster results; the physical prototype of the collecting robot.

References

[1] M. Amar, M.U. Asad, U. Farooq, A. Raza, A. Iqbal and O. Tahir, Ball-Scorer: design and fabrication of an autonomous ball scoring robot, In: *The 2nd Intern. Conf. on Computer and Automation Engineering (ICCAE)*, **1**, IEEE, Singapore (2010), 598-602; DOI: 10.1109/ICCAE.2010.5451341.

[2] M.H. Coen, Design principles for intelligent environments, In: *The Fifteenth National Conf. on Artificial Intelligence (AAAI-98)*, Madison (1998), 547-554.

[3] E.R. Dougherty and R.A. Lotufo, *Hands-on Morphological Image Processing*, SPIE Publications, Washington (2003).

[4] M. Dunbabin, J. Roberts, K. Usher and P. Corke, In the rough: in-field evaluation of an autonomous vehicle for golf course maintenance, In: *IEEE/RSJ Intern. Conf. on Intelligent Robots and Systems (IROS 2004)*, **4**, IEEE, Sendai (2004), 3339-3344; DOI: 10.1109/IROS.2004.1389932.

[5] U. Farooq, H. Khan, A. Mateen, M. Amar, M.U. Asad and A. Iqbal, Design and development of an autonomous ball potting robot for NERC Contest, In: *2nd Intern. Conf. on Advanced Computer Control (ICACC)*, **4**, IEEE, Shenyang (2010), 394-398; DOI: 10.1109/ICACC.2010.5486895.

[6] F. Glover, Scatter search and star-paths: beyond the genetic metaphor, *Operations-Research-Spektrum*, **17**, No 2-3 (1995), 125-137.

[7] F. Glover, Tabu search and adaptive memory programming: Advances, applications and challenges, In: R. Barr, R. Helgason and J. Kennington (Eds.), *Interfaces in Computer Science and Operations Research*, Kluwer, Boston (1996), 1-75; PMid:8842811.

[8] R.C. Gonzalez and R.E. Woods, *Digital Image Processing*, Prentice-Hall, Upper Saddle River (2006).

[9] R.M. Jusoh, Application of vision target localization for mobile robot, In: *2006 4th Student Conf. on Research and Development (SCOReD 2006)*, IEEE, Selangor (2006), 144-146; DOI: 10.1109/SCORED.2006.4339327.

[10] H. Nagahashi, A. Niwa and Takeshi Agui, Competition and mutualism in a simulation of adaptive artificial organisms, In: *IEEE Intern. Conf. on Evolutionary Computation (ICEC 1995)*, **2**, IEEE, Perth (1995), 695-700; DOI: 10.1109/ICEC.1995.487469.

[11] P.M. Pardalos and M.G.C Resende, *Handbook of Applied Optimization*, Oxford University Press, New York (2002).

[12] A. Prati, I. Mikic, M.M. Trivedi and R. Cucchiara, Detecting moving shadows: algorithms and evaluation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**, No 7 (2003), 918-923; DOI: 10.1109/TPAMI.2003.1206520.

[13] R.J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, Image change detection algorithms: a systematic survey, *IEEE Transactions on Image Processing*, **14**, No 3 (2005), 294-307; DOI: 10.1109/TIP.2004.838698.

[14] S. Savage, Statistical indicators of optimality, In: *IEEE Conf. Record of 14th Annual Symposium on Switching and Automata Theory (SWAT'08)*, IEEE, Iowa City (1973), 85-91; DOI: 10.1109/SWAT.1973.26.

[15] M. Srinivas and L.M. Patnaik, Genetic algorithms: A survey, *Computer*, **27**, No 6 (1994), 17-26; DOI: 10.1109/2.294849.

[16] Zhiqiang Ma and Hyongsuk Kim, Autonomous technologies of global vision-based golf ball collection robot, In: *2011 IEEE International Conference on Automation and Logistics (ICAL)*, IEEE, Chongqing (2011), 497-500; DOI: 10.1109/ICAL.2011.6024770.