Available online at www.sciencedirect.com
ScienceDirect
Expert Systems with Applications 36 (2009) 3153–3166
Expert Systems with Applications
www.elsevier.com/locate/eswa

Web-based mobile robot platform for real-time exercises

Seref Sagiroglu [a,], Nihat Yilmaz [b]

[a] Department of Computer Engineering, Faculty of Engineering-Architecture, Gazi University, Maltepe, 06570 Ankara, Turkey
[b] Department of Electrical-Electronics Engineering, Faculty of Engineering-Architecture, Selcuk University, Konya, Turkey

Abstract

This paper introduces a new vision-based and web-based mobile robot platform. The platform consists of control and communication centers, a mobile robot and real-time support libraries. All activities in the platform are achieved by only computer vision techniques. The platform provides monitoring, tele-controlling and programming for real-time educational exercises and helps to the users to achieve these exercises through a standard web browser without any need for additional support software. The results have shown that the proposed, designed and implemented platform provide amazing new facilities and features to the users (students and researchers) in applying their real-time exercises on web.

Keywords: Web-based; Real-time; Exercises; Platform; Design

1. Introduction

Recent developments in web technologies force traditional approaches to be supported by new strategies, approaches and challenges in many applications (Guimaraes et al., 2005; Halme, Leppanen, Suomela, Ylonen, & Kettunen, 2003; Hu, Yu, Tsui, & Zhou, 2001; Jacobsen et al., 2004; Marin, Sanz, & Del Pobil, 2003; Saucy & Mondada, 2000; Schilling, Roth, & Splica, 2005; Stein, 2000). In those, remote experiments have been mostly achieved. Remote and web-based laboratories are literally exploding new techniques and approaches increasingly adopted for education. First generation of web-based robot laboratories, named as tele-robot is mainly based on manipulators (robotic arms) or simple mobile robots that are directly controlled by human operators while second generation of web-based robot laboratories operates for uncertain environment and autonomic activities (Oboe, 2001; Robinette & Manseur, 2001; Rosch, Schilling, & Roth, 2002). The key features of second generation web based robotic laboratories are their adjustable or programmable structures, which enable them to be used for educational and scientific purposes in the real-world environments (Kuc, Jackson, & Kuc, 2004; Patel, Sanyal, & Sobh, 2006; Pipe & Carse, 2007; Schilling et al., 2005; Simmons, Fernandez, Goodwin, Koenig, & O'Sullivan, 2000). The mobile robots used in second generation robot laboratories have been equipped with touch sensors, range meter sensors, proximity sensors, contrast sensors, cameras, and sound synthesizer for realizing activities automatically. Thus, the information acquired from sensors can be used to monitor results or provides feedback for control or other purposes. The cameras are used for monitoring the robot actions. But, direct use of camera for robot motion control is seldom at web-based laboratories. The reasons are that vision-based web-robot applications require real-time control, real-time image processing, high bandwidths to transfer raw or processed images to the user platform and a human operator; it might also cause delay in operation (Smith & Hashtrudi-Zaad, 2006; Trahanias et al., 2005), programs for robot vision can not be changed through web interface (Kwon, Rauniar, Chiou, & Sosa, 2006; Marin et al., 2003).

Corresponding author. Tel./fax: +90 312 2306503.
E-mail addresses: ss@gazi.edu.tr (S. Sagiroglu), nyilmaz@selcuk.edu.tr (N. Yilmaz).

0957-4174/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2008.01.046

In this study, a web platform supported by a mobile robot system for real-time experiments has been designed and implemented. The robot system especially designed for even surfaces supports the activities on R& D, observation, exploration, security and especially courses in engineering education. The designed platform named as Web-based Small Universal Navigator (Web-SUN) accomplishes various tasks through web-based communication and control units. Some of these units have been introduced in Sagiroglu, Yilmaz, and Bayrak (2005, 2006), Yilmaz (2005), Yilmaz, Sagiroglu, and Bayrak (2006a, 2006b). Web-SUN provides a fully operated platform to users to support real-time robot vision, control and programming exercises. The platform has a mobile robot to be controlled remotely by the developed hardware and software without using any sensor except camera and optical tachometers. This makes the system distinct from the similar studies presented in the literature.

This paper is organized as follows. Section 2 describes the structure of web-based real-time platform presented in this work. Section 3 gives the design and implementation details of Web-SUN. In Section 4, some of the works can be achieved on the platform were introduced. Finally, experimental results were presented and concluded in Section 5.

2. Proposed system (web-SUN) or platform

General structure of the proposed system or platform (Web-SUN) is demonstrated in Fig. 1. The system mainly consists of web portal and control center including mobile robot vehicle, user and database management modules, support software, wireless access point, communication center, administration, support libraries, control center and portable PC were also parts of the platform. Web-SUN (Small Universal Navigator) was mainly designed to build up a multi-functional real-time web exercises.

In general, the platform involves in integrating three major disciplines, namely, mechanical, electronic and software engineering to support online or real-time services for desired applications. It needs to be emphasized that most of the systems having hardware, software and mechanics were developed by the authors. Web-SUN system having a three-wheel-robot vehicle, a two-degrees-freedom camera motion system and a web-based control center were used to achieve real-time experiments. A portable PC was mounted on the robot vehicle to support all processes achieved within the mobile platform. In order to support Web-SUN system and reduce the operation time during real-time processes, various real-time applications with the help of support library were developed in this work. All activities achieved within Web-SUN platform can be monitored, controlled and watched on-line by the users through the designed web platform.

The hardware used in the platform was comprised of a three-wheel mobile robot system, a two-degrees-of-freedom camera motion unit, a portable PC, various controllers and wireless communication modules. The software developed for the platform covers the programming for microcomputer, control center and web interface. Details of the designed Web-SUN platform parts are given in the following section.

2.1. Structures of mechanics and hardware

A three-wheel-robot vehicle, a two-degrees-freedom camera motion system and control units designed and implemented in this work were shown in Fig. 2. The portable PC supports movements, communication, on-line services and various controllers. The PC could not be seen in the figure as it was built in the robot body.

The robot vehicle consists of microcontrollers, a camera system, a portable PC with Wi-Fi LAN adapter, motors and motor drives used to carry out vehicle and camera movements and speed sensors used to estimate the position of it. As mentioned earlier, the robot is a three-wheel designation of which rear wheels are free to move. The front wheel steers and drives the mobile robot. For the simplicity

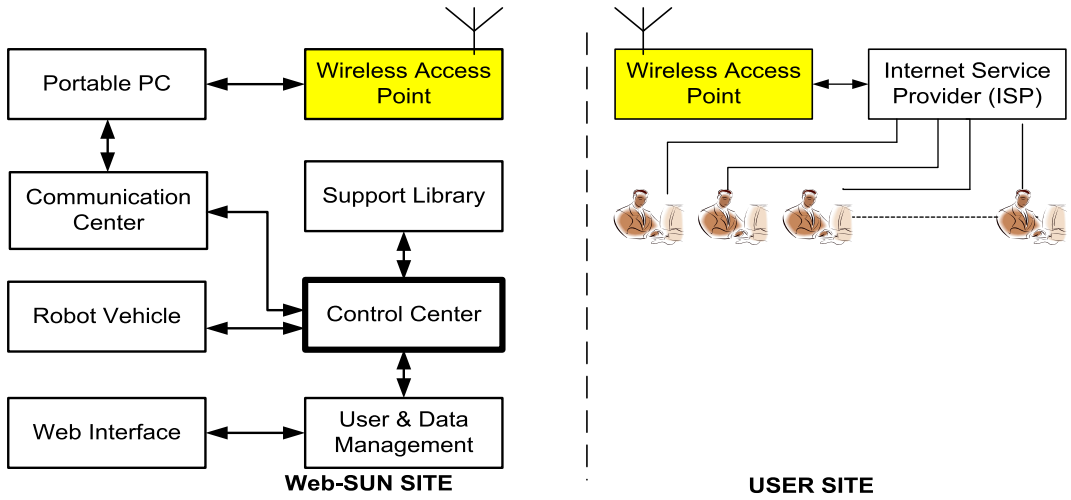


Fig. 1. General structure of Web-SUN system.

Fig. 2. Part of Web-SUN platform: mobile Robot vehicle.

and the cost, the body of robot vehicle was constructed from PVC (polyvinylchloride) and polyethylene materials.

Two microcontrollers (the slave and the master) were employed in hardware implementation. The slave microcontroller controls the camera system, the motors and the sensors and provides communication with master microcontroller via I^2C serial bus. The master microcomputer manages the vehicle motion control, the communication with portable PC and the slave microcontroller. The master microcontroller was connected to the PC via RS232 port. In addition, four DC motors with gearbox, four optical tachometers coupled to the DC motors, four H-bridge inverters for DC motor drive and two LCDs were used for this implementation. The control center was designed to manage all activities of the robot vehicle by the users easily.

Details of the kinematics equations and drive systems of Web-SUN were given in Yilmaz (2005), Yilmaz et al. (2006a). Positions of Web-SUN were computed using the kinematics equations by the master microcontroller. PID controllers were used to control the positions for vehicle and camera motion systems. The controllers were individually run on master and slave microcontrollers.

Actuations of the camera were implemented using two DC motors. The motors turning 360° horizontally (pan) and 270° vertically (tilt) were selected. Motion coming from the first DC motor is transferred to the vehicle with a gear so that horizontal pan motion was provided. The motion coming from the second DC motor tilts the robot vehicle vertically via a gear set. The DC motors were driven with PWM based H-Bridge DC motor drivers controlled by PIC16F877.

Intel Centrino 1.6 GHz microprocessor, 512 MB RAM and 30 GB HDD and internal Wi-Fi LAN adapter was used as a PC.

2.2. Software implementation

Web-SUN platform serves many users through internet with the help of developed software as shown in Fig. 3. In order to achieve the processes in Fig. 3, internet software for browsing, server software for http, main program for control center and its individual parts, robot vehicle and camera control programs for PIC microcontrollers and database programs for authentication, verification and storing were developed for real-time web platform exercises. As can be seen from the figure, active and passive users facilitate from the platform according to the tasks and expectations. In the system, the active user operates the mobile robot through the web browser or manages all processes available on the web interface. The other (passive) users watch the operations implemented by the active

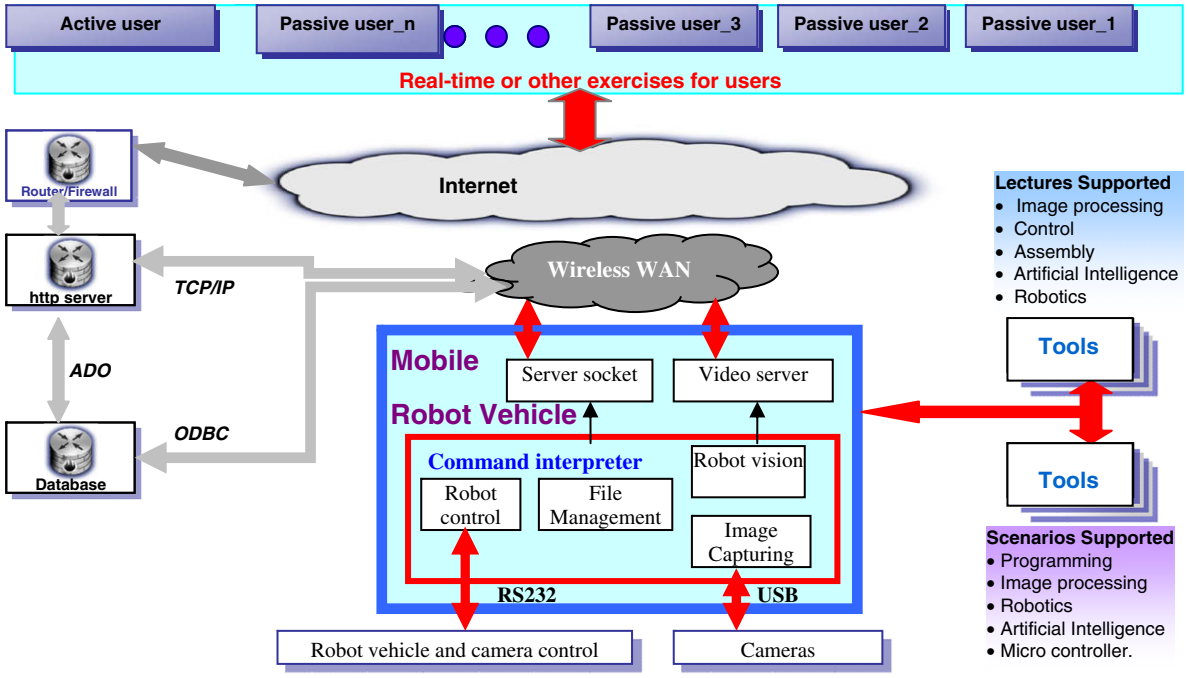


Fig. 3. Software structure of Web-SUN platform.

user. All users connected to the platform can communicate with each other by means of a messaging service and a shared working environment be configured.

The programs developed in this study mainly consist of three parts including microcontrollers, control center and web interface. Programming the web applications was based on PHP for database and JavaScript for timing and refreshing functions. The microcontrollers were programmed using MPLAB assembler programming. The control center software was developed in Delphi as depicted in Fig. 4. Capturing images and serial communication were acquired via the third party components. Web-SUN control center used in web interface was given in Fig. 5. Control Center Program was the most important component of Web-SUN and helped to manage the system smoothly, to execute the programs, to facilitate the users form support libraries, to develop and test new studies, to utilize user privileges of the control center.

Communications among the wireless access point (WAP) and the Wi-Fi LAN adaptor, the control center and the master microcontroller, and the master and the slave microcontrollers were achieved with the help of the designed and developed software modules. Initial communication between the control center and internet service provider (ISP) was realized wireless using a WAP and Wi-Fi LAN adapter. Communication range was expanded to 1 km using an external active antenna. The other communication between the control center and the master microcontroller was provided through RS232 port in the form of Modbus ASCII, 19200 Baud, 8 bit and odd parity. The last communication between master and slave microcontroller was acquired through I^2C serial port.

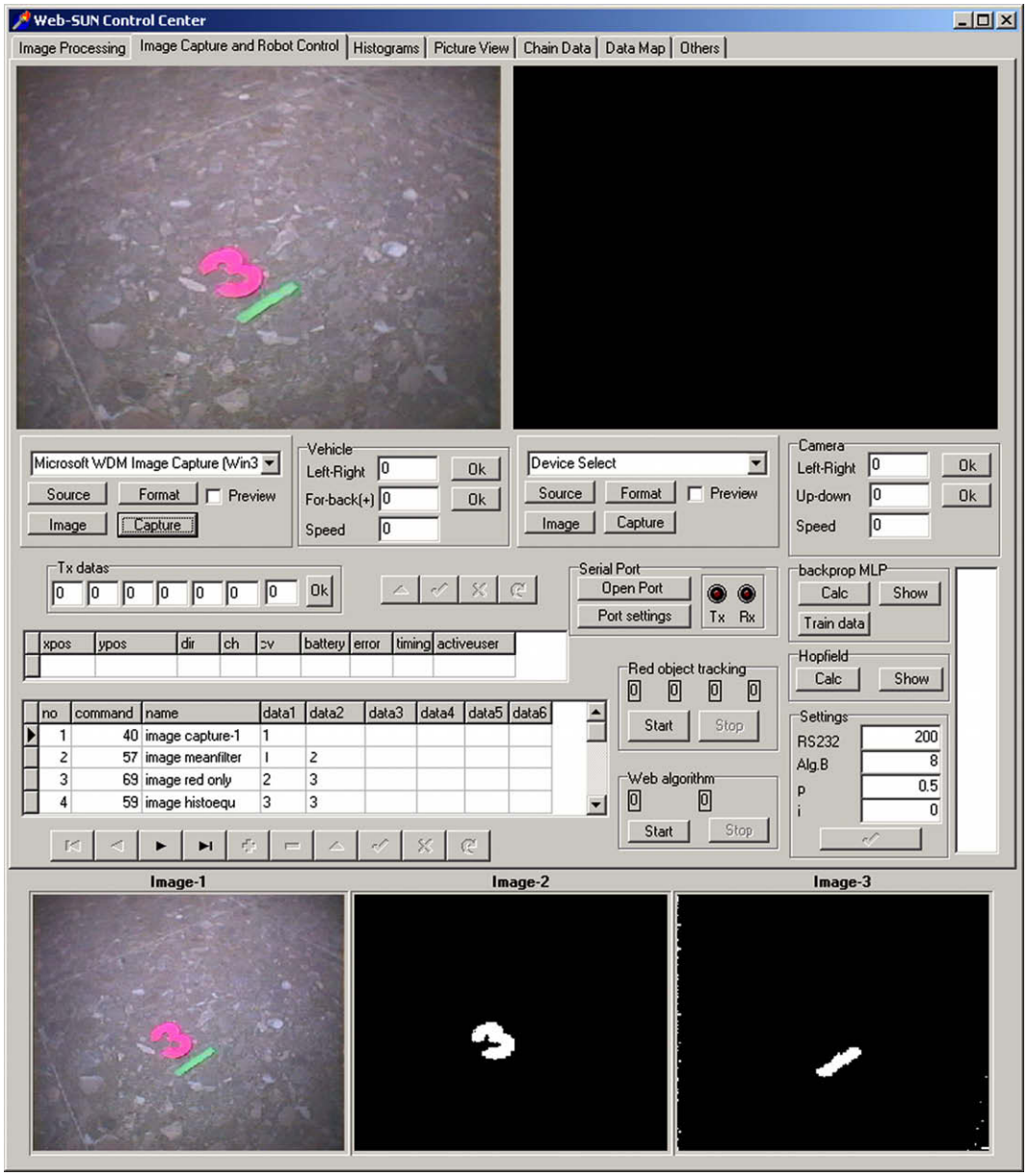


Fig. 4. A screenshot for Web-SUN control center.

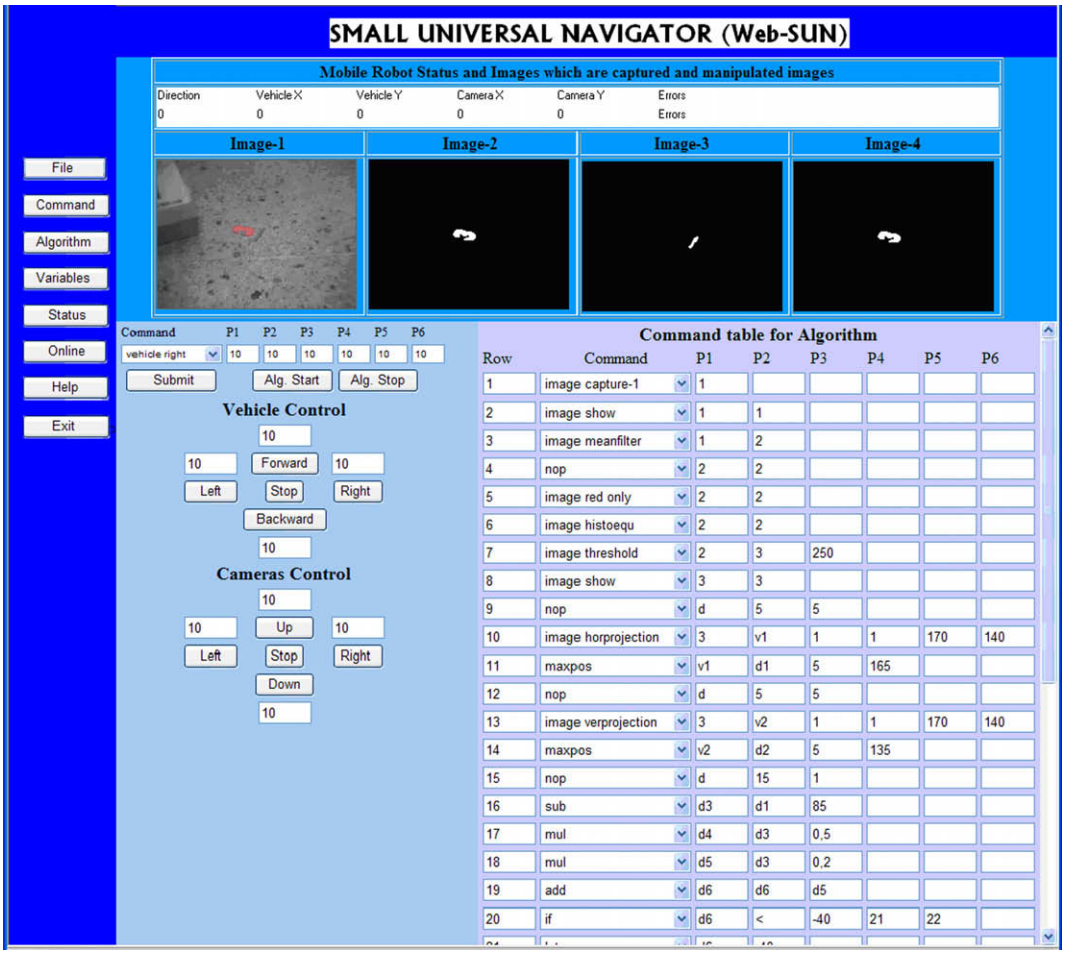


Fig. 5. A screenshot from Web-SUN interface software.

The programs were also developed for a smooth communication between the mobile robot and the web interface used by users, interpretation and execution of programs written through the web interface, and transferring results (images and variables) to the web interface. The control center program was supported with standard IIS web server and database software.

The control center executes control programs running on the computer and communication units. The control center mainly realizes video and web services, tele-control, vehicle positioning and communication, program interpretation, vision-based autonomous or semi-autonomous robot control and real-time application development. Web server supports web portal for the users of Web-SUN. Video server and *program interpreter* run the image based robot program and monitor the desired results achieved from real-time implementations. Tele-control responses for immediate commands coming from a web user and determines the system status. Communication and determination of vehicle position are managed to communicate with the robot vehicle. Position of robot vehicle can be computed from the data achieved from the information encoders. For vision-based autonomous or semi-autonomous robot control, the images captured from the camera were processed and the required knowledge is extracted for a human operator or autonomous robot control algorithm. In order to support the users for real-time exercises, a module was also developed to communicate with the support libraries.

Control center mainly helps to achieve both simple and complex communications tasks between master microcontroller and control center. Simple tasks were to turn right or left, to move backward or forward and to adjust speed of camera and vehicle motions. Complex tasks required for autonomous operations were wall and line tracking, object tracing, localization and finding landmarks, etc. These tasks can be loaded to the control center via a web page on internet. All commands for the tasks were first stored to the system database and then executed. This reduces the process time as it was a vital concern to execute several tasks simultaneously. While one task continues the other starts so enough time is required to distinguish the tasks. The status of tasks is also stored to the database. It needs to be emphasized once more that all complex tasks were realized by the control center.

The other part of Web-SUN was the communication center. This center provided wireless internet connection to the robot having a web server and a wireless access point

on it. Windows XP based standard server programs were run in the center to achieve different real-time exercises. Image processing and artificial neural network libraries were developed to create the applications in the control center software. The users can develop different applications with the help of these libraries. The libraries were briefly introduced in this article in Section 3.

Web interface utilities were important components for the system. Applications were developed from a remote place via the web interface independently as shown in Fig. 5. Active and passive users had to register to the system. While an appointment was required for the active user, passive users do not require appointments as they only monitor or trace the activities operated by the active users.

Two methods were followed in the operations to result the experiments on the system via web interface. First method was to result the command immediately and to view its results. Tele-controlling processes (forward, backward, left and right button for camera and robot vehicle) and the processes using libraries were realized in the first method. In addition, a combo-box, six edit boxes and three buttons supported tele-control processes. The processes were executed depending on the desired command selected from a combo-box. The parameters related with this command were entered to edit boxes in an order appropriately. The desired command is then executed via a submit button on the web platform. Second method was preferred for complex processes, semi-autonomous or autonomous robot controls. This method enabled users to write or develop computer programs through web interface. An algorithm environment was created to do this operation. It was then converted to program codes on web interface. Any program for an operation can be written step by step with the help of "Command table for algorithm" button on the web interface. In order to provide faster computation and a user-friendly environment to the users during programming, library functions were designed for Web-SUN platform. There were step numbers for every command which was selected via a combo-box. The parameters related with this command can be written to the edit boxes provided. Help file for commands in use and examples of program codes can be called with "Help" button. Through the web interface, the programs can be executed with the available buttons on the menu screen.

Results (images and variable values) obtained from immediate command running and execution of written programs were shown at the top of web interface. When "status" button was clicked, robot position values and images from the camera were monitored. When "Variables" button was clicked, the parameter values used in program were monitored on the web interface.

"File" and "Online" buttons on web interface were used for uploading file to the system by a user and monitoring live video stream of robot move, respectively. Besides video monitoring, chat facilities were also provided for the users at the same frame. These facilities were in the same frame with "Command table for algorithm" section on web interface. To return to the algorithm section, "Algorithm" button requires to be clicked.

3. Support libraries for web-SUN platform

As mentioned earlier, the libraries were designed to support the users to do their image processing, artificial neural networks and control exercises easily. These libraries were explained below.

3.1. Image processing

Image space transformations, arithmetic and logic operators, filters, edge detectors and feature extraction techniques were developed within this library to support real-time image processing and robot vision courses. For online or real-time web exercises, these functions were progressively developed and installed into the web-based mobile robot system. With the help of this support library, simple and complex vision sensing problems can be solved easily and effectively on the web environment for various applications.

An active user can use ready functions or also employ his/her own function and DLL function. It is also possible to add a new library function to the system by uploading it with the permission of administrator. At present, adding any additional function to the system requires restarting the PC on the platform.

Interface software was developed to assist the system administrator and developers. The software based on windows operating system can provide an efficient interface media tools to the users on the platform. Therefore, the effective image processing tool can be used for processes apart from the tools developed for the platform. The developed interface shown in Fig. 6 was composed of filters, edge detection operators, point operators, arithmetic operators, color space conversions, transformations, logic operators, geometric operators, histograms, projections and statistical analysis tools. Gray tone, histogram, histo, picture open and picture save buttons placed at the top of the interface and conversion of colored images into gray toned images, histogram equalization, picture opening from file and picture saving to file have been achieved using the interface given in Fig. 6.

Convolution processes were realized after selecting Convolution kernel button. Ten different convolution kernels were available for the users to be used in different applications. However, a special kernel can be inserted to the program through a 3×3 string grid. At the color conversation panel, there were 16 buttons to obtain each part of RGB, HIS, Lab and XYZ color space components.

At the transformations panel, Hough and FFT transformations, and chain code production can be realized by using Hough, FFT and Chaincode buttons.

Point operators, different threshold processes, labeling, smearing, thinning and enlarging can be applied to images

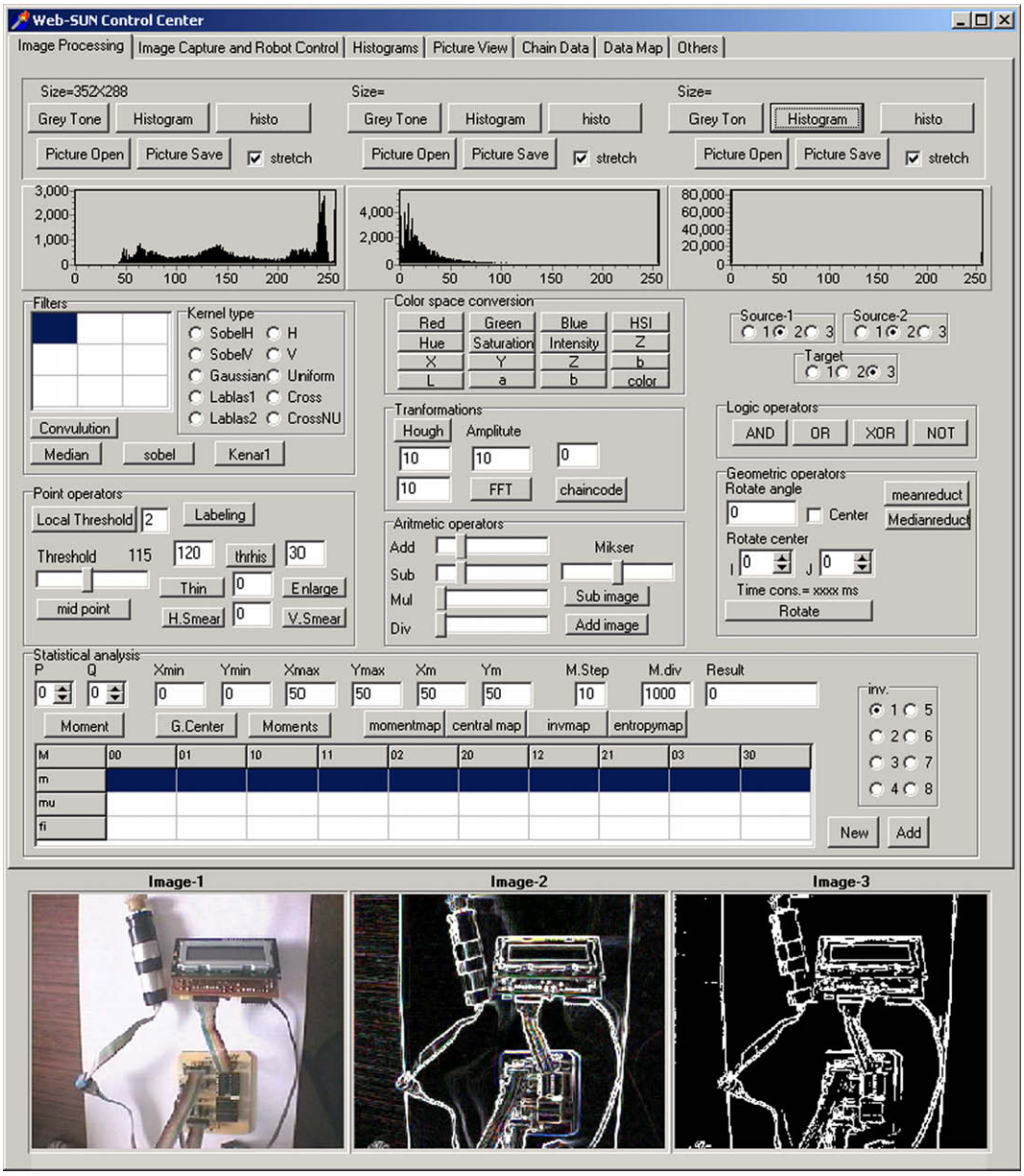


Fig. 6. Screenshot of image processing part of control center program.

within the provided media. Arithmetic add, subtract, multiplication and division processes can be achieved among numbers and images via the arithmetic operation frame. However, addition and subtraction processes can be made between two images.

AND, OR, XOR and NOT logical processes can be also realized via buttons on the logical operators panel. At the media of geometric operators, there were some buttons related with rotation and dimension reduction processes. Statistical processes (moments, central moments, invariance moments, gravity center processes, etc.) can be also accomplished by objects on statistical analysis media.

While using image processing feature of the system, in addition to the real workplace images acquired from robot's camera, the users can also work with their own images by uploading them to the system.

All of these functions and other extended image processing functions were designed to be used on web for robot control and other real-time exercises. As a result of this, web interface was the basic application development environment. All of image processing functions can be executed immediately on the web with the help of combo-box and "Submit" button on web interface as shown in Fig. 5. Furthermore, these functions can be used for programming or reprogramming tasks on web via "Command table for algorithm" menu.

3.2. Artificial neural networks

Artificial neural networks (ANNs) are biologically inspired intelligent techniques. They have generally a number of simple and highly interconnected neurons organized

into layers. ANNs have many structures and architectures (Haykin, 1994; Maren, Harston, & Pap, 1990). Multilayered perceptron neural networks (MLPNNs) are the simplest and therefore most commonly used neural network architectures (Haykin, 1994). An MLPNN has mainly three layers: an input layer, an output layer and an intermediate or hidden layer. Neurons in the input layer only act as buffers for distributing the input signals x_i to neurons in the hidden layer. Each neuron j in the hidden layer sums up its input signals x_i after weighting them with the strengths of the respective connections w_{ji} from the input layer and computes its output y_j as a function f of the sum

$$y_j = f\left(\sum w_{ji}x_i\right) \tag{1}$$

where f can be a sigmoid or hyperbolic tangent function. The output of neurons in the output layer is computed similarly.

Training a network consists of adjusting weights of the network using a learning algorithm. The back-propagation with or without momentum (BPM) learning algorithm (Rumelhart & McClelland, 1986) is used in this work because of being most commonly adopted MLPNN training algorithm. It is a gradient descent algorithm and gives the change $\Delta w_{ji}(k)$ in the weight of a connection between neurons i and j as follow:

$$\Delta w_{ji}(k) = \alpha\delta_j x_i + \mu\Delta w_{ji}(k-1) \tag{2}$$

where x_i is the input, α is the learning coefficient, μ is the momentum coefficient, and δ_j is a factor depending on whether neuron j is an output neuron or a hidden neuron. For output neurons,

$$\delta_j = \frac{\partial f}{\partial \mathrm{net}_j}(y_j^T - y_j) \tag{3}$$

where $\mathrm{net}_j \equiv \sum x_i w_{ji}$ and y_j^T is the target output for neuron j. For hidden neurons,

$$\delta_j = \frac{\partial f}{\partial \mathrm{net}_j}\sum_q w_{qj}\delta_q \tag{4}$$

As there are no target outputs for hidden neurons in Eq. (4), the difference between the target and actual output of a hidden neuron j is replaced by the weighted sum of the δ_q terms already obtained for neurons q connected to the output of j. Thus, iteratively, beginning with the output layer, the δ term is computed for all neurons in all layers except input layer and weights were then updated according to Eq. (2).

Afore mentioned the platform presented in this work has a neural network library to be used for different applications such as identification, recognition, classification or analysis. MLPNN structures were also supported within this platform with the help of neural library developed. The web users can use this neural library on their robotic, control, programming and image processing applications. Training and test processes in ANNs and their parameters can be set to the required values on the web menu provided. The program interface used for this purpose was shown in Fig. 7. All ANN parameters can be selected or adjusted according to the user requirements. For a real-time web robot application, the trained neural network parameters can be uploaded to the web server. It is then used for the application.

Generally, the input data sources to the ANN might be images or features extracted from the captured images. The images were pre-processed with the help of the functions in image processing library, which were designed for Web-SUN as well.

3.3. Control

Control actions in the system were performed in two ways. In first, control algorithms had to be inherently present in the system for fully robotic control activities and the programs were then executed without requiring libraries (or library functions) with simple commands written on web portal. Another way to use control algorithms is to integrate the algorithms such as PID or Fuzzy controller. The robot camera is the primary information source and has a important role on the control of Web-SUN. The control of robot motion was achieved from the information acquired from the image source and optical feedback elements present on geared DC motors actuating on Web-SUN. These feedback signals were used to implement and control the speed, position and acceleration of the system. The robot vehicle control could be implemented by utilizing all these input information, either using programs developed by the user with basic commands or using programs employing conventional control algorithms. The source of program code was typed into the web page of Web-SUN with the help of web browser. They were then loaded to the system and executed. The results were finally presented and analyzed with the help of support media.

4. Using web-SUN platform for real-time landmark detection experiments

As mentioned earlier, Web-SUN platform was designed for various purposes such analysis, navigation, recognition, observation, detection and line tracking. In this article, landmark detection, one of web-based real-time experiments for Web-SUN, was considered. For this purpose, two types of landmarks were used to navigate the mobile robot. First, landmarks were used to correct the position of mobile robot. In order to do that a workplace were designed in the laboratory. A number of metal number signs underlined were prepared to achieve this task. The numbers (in red) having underlined (with green) were recognized by Web-SUN with the help of the images captures from the ground. The underline (green) helps the robot to correct position. It needs to be reminded that the real positions of the numbers are known by the user.

The second type of landmarks was employed for map creation. The landmarks were natural objects such as flow-

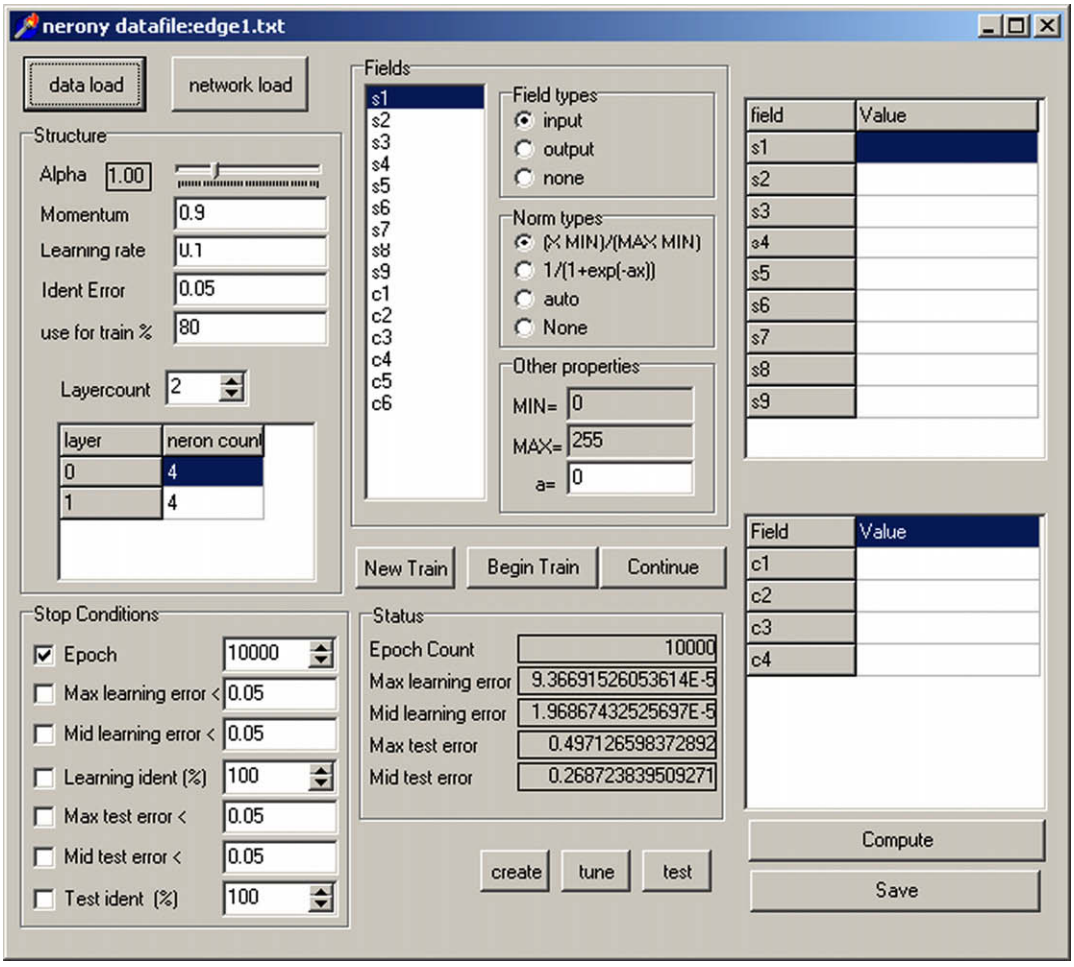


Fig. 7. A screenshot for neural network library.

ers, wall, ground and wall edges. Locations of these natural objects were calculated according to the mobile robot position. Locations were recorded on a map for further processes. We define positioning as a process that uses the sensory input of the robot to derive the location and orientation of the robot with respect to the map of the environment. Image source for land-marking was achieved from the camera (webcam) mounted on the mobile robot. The captured image was RGB color image having 320×240 resolutions.

4.1. Searching landmarks

Searching processes on the captured image can be achieved for different tilt angle of the camera as shown in Fig. 8. A histogram-based scanning algorithm was used

(a) image #1 (b) image #2

Fig. 8. Images for landmark scanning processes.

in this work. Captured images were scanned via variable size according to searching windows for colored (red and green) landmarks.

4.2. Pre-processing and feature extraction

Pre-processing was achieved after finding the landmarks roughly as shown in Fig. 8. Pre-processing consisted of median filter and histogram equalizations. Thus, the pre-processed image was ready for feature extraction process. There were a number of alternatives for feature extraction processes consisting of image raw data, histogram, projections and statistical properties of these. In this paper, invariant moment values of image raw data were used. The invariant moments were calculated using moments, central and normalized central moment values. Equations for invariant moments were given in Eqs. (5)–(16). Moment values of an image can be calculated from

$$m_{i,j} = \sum_{x=0}^{m}\sum_{y=0}^{n} x^i y^j P(x,y) \tag{5}$$

where $P(x,y)$ is the brightness values of image at x and y coordinates. Central moment values of an image can be computed as

$$\mu_{i,j} = \sum_{x=0}^{m}\sum_{y=0}^{n} (x-c_x)^i (y-c_y)^j P(x,y) \tag{6}$$

where c_x and c_y are the coordinates of gravity center of image and they are computed from

$$c_x = \frac{m_{1,0}}{m_{0,0}} \quad c_y = \frac{m_{0,1}}{m_{0,0}} \tag{7}$$

The normalized central moment values can be also obtained from

$$\eta_{i,j} = \frac{\mu_{i,j}}{\mu_{0,0}^{\gamma}} \tag{8}$$

where γ is $\gamma = (i+j)/2+1$. As a result, seven invariant moment values were derived from Eqs. 5, 7 and 8 as given in Eqs. (9)–(15).

$$\varphi_1 = \eta_{2,0} + \eta_{0,2} \tag{9}$$

$$\varphi_2 = (\eta_{2,0} - \eta_{0,2})^2 + 4\eta_{1,1}^2 \tag{10}$$

$$\varphi_3 = (\eta_{3,0} - 3\eta_{1,2})^2 + (3\eta_{2,1} - \eta_{0,3})^2 \tag{11}$$

$$\varphi_4 = (\eta_{3,0} - \eta_{1,2})^2 + (\eta_{2,1} - \eta_{0,3})^2 \tag{12}$$

$$\begin{aligned}\varphi_5 = {} & (\eta_{3,0} - 3\eta_{1,2})(\eta_{3,0} + \eta_{1,2})[(\eta_{3,0} + \eta_{1,2})^2 - 3(\eta_{2,1} + \eta_{0,3})^2] \\ & + (3\eta_{2,1} - \eta_{0,3})(\eta_{2,1} + \eta_{0,3})[3(\eta_{3,0} + \eta_{1,2})^2 \\ & - (\eta_{2,1} + \eta_{0,3})^2]\end{aligned} \tag{13}$$

$$\begin{aligned}\varphi_6 = {} & (\eta_{2,0} - \eta_{0,2})[(\eta_{3,0} + \eta_{1,2})^2 - (\eta_{2,1} + \eta_{0,3})^2] \\ & + 4\eta_{1,1}(\eta_{3,0} - \eta_{1,2})(\eta_{2,1} + \eta_{0,3})\end{aligned} \tag{14}$$

$$\begin{aligned}\varphi_7 = {} & (3\eta_{1,2} - \eta_{3,0})(\eta_{3,0} + \eta_{1,2})[(\eta_{3,0} + \eta_{1,2})^2 - 3(\eta_{2,1} + \eta_{0,3})^2] \\ & + (3\eta_{2,1} - \eta_{0,3})(\eta_{2,1} + \eta_{0,3})[3(\eta_{3,0} + \eta_{1,2})^2 \\ & - (\eta_{2,1} + \eta_{0,3})^2]\end{aligned} \tag{15}$$

These seven features extracted for red, green and blue colors using invariant moment equations were used as inputs to neural network recognition units.

4.3. Neural network based recognition

In this study, MLPNN structure was used to recognize the landmarks with the help of support library designed for Web-SUN as explained earlier. During recognition processes on the web, the ready commands available in the library were used for training and test phases. In this study, the neural network was trained with the back-propagation learning algorithm.

The recognition processes were achieved with the help of two neural network modules. The first module considers the recognition of the land-marked numbers as shown in Fig. 8. The outputs of this module were used to correct position of the mobile robot. The module was trained with 60 real data sets extracted from the land-marked images. The second module achieves other recognition tasks consisting of flower, wall, ground and other objects. The outputs of this module were used for mapping the environment. The module was trained with 115 real data sets extracted from other landmark images. Some of images used in training different ANNs were illustrated in Fig. 9. It needs to be emphasized that those images were used in training after pre-processing. All data sets for different landmarks were obtained from the vision system of Web-SUN experimentally.

It needs to be declared that all variables were also scaled before training. The data used to train and test the neural network models were subdivided into two sets. In the first test, neural network module having 48 observations was used for the position correction. In order to test the module 12 different observations were used. In the second test, neural network module having 98 observations was used for mapping the environment. For this test, 17 different observations were used.

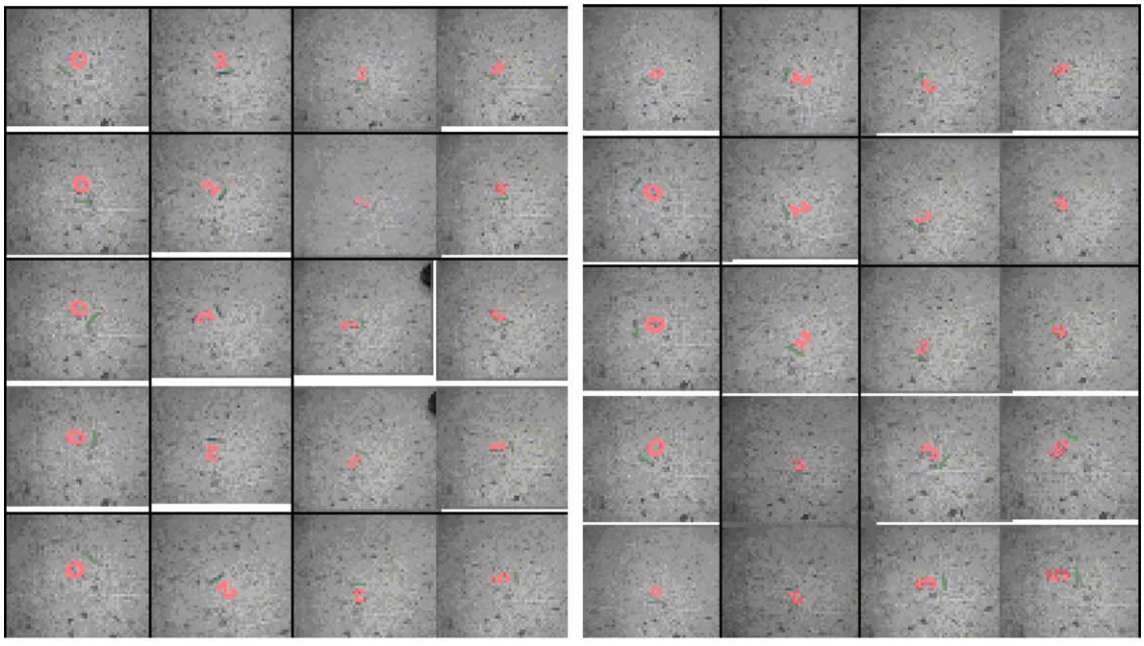
The structures of ANNs (number of layers, number of hidden nodes, learning rate and momentum values) were optimized during the training phase to guarantee the convergence for landmark recognitions.

After training, the performance of each ANN module was evaluated according to the following formula:

$$\text{Error}(\%) = \left(\frac{\sum_{i=1}^{k} |d(i) - a(i)|}{m^* n} \right) * 100 \tag{16}$$

where $d(i)$ is the desired output, $a(i)$ is the ANN output, k is the number of samples in training or test data, m is the number of segments used in training or test data sets and n is the number of ANN outputs. In addition, R-Square Correlation (RSC) measure as formulated in Eq. (17) was also used to determine an optimum ANN structure.

$$\text{RSC} = r^2 \tag{17}$$



(a) Training set #1

(b) Other data sets used in training

Fig. 9. ANN training sets used for various landmarks.

where

$$r = \frac{\sum (x-\bar{x})(y-\bar{y})}{\sqrt{\sum (x-\bar{x})^2 \sum (y-\bar{y})^2}}$$

4.4. Operation steps

Operation is started with training ANN structure. This operation can be done either downloading the ANN software into a PC from the web platform or upload training data sets into Web-SUN portal to make training via "built-net", "trainnet" and "testnet" commands, which were not given in the figures. After proper training, the ANN was used to detect landmarks with the help of the algorithm designed for this purpose. The algorithm for landmark detection was given below as:

(1) net1 = loadnet (landmark data_1)
(2) net2 = loadnet (landmark data_2)
(3) result = scan_landmark (camera_1, image_2)
(4) if result = false then Goto (3) else Goto (5)
(5) array1 = feature_extract(image_2, 64)
(6) array2 = testnet(net1, array1)
(7) object_name = analysis_test_results(array2)
(8) if object_name = nil then Goto (10) else Goto (9)
(9) object_place_ to_ map (cam_angles, robot_position, object_name)
(10) array3 = testnet (net2, array1)
(11) landmark_name = analysis_test_results (array_3)

(12) if landmark_name = nil then Goto (14) else Goto (13)
(13) robot_position_correction (cam_angles, robot_position, landmark_real_positions (landmark_name))
(14) show (array_2)
(15) show (array_3)
(16) Goto (2)

After these steps, the realization on Web-SUN was achieved as

1. The weights and the structure of a well designed of ANN module are uploaded to the system as a file via "FILE" button.
2. the algorithm prepared for the experiment is loaded step by step via "algorithm" button.
3. The algorithm loaded is executed via "Alg.Start" button.
4. The executed algorithm is stopped via "Alg.Stop" button.

5. Experiments

The results were considered for three subsections. As mentioned earlier, two of them were about land marking. The third one was about to show other capabilities of the Web-SUN.

5.1. Landmark detection for numbers on ground

One of the landmark examples was to recognize numbers on the ground. 21 features were extracted for red, green and blue pairs of landmark images using invariant moment equations given in Eqs. (5)–(15). These features were used as inputs to the designed neural network recognizer. In this study, a MLPNN structure was used for landmark recognition as mentioned earlier. The ANN structure has 21 neurons in the input layer, 8 neurons in the both hidden layer, and 5 neurons in the output layer. The structure was trained with the back-propagation with momentum learning algorithm. In training, the number of iterations was set to 12,000, the learning and the momentum coefficients were fixed to $\alpha = 0.9$ and $\mu = 0.7$, respectively. The ANN structure was initially trained with 48 data sets and then tested with 12 different data sets. The errors for number recognition in training and test were 0.00013% and 0.026%, respectively. The correlation was $r^2 = 0.99$.

5.2. Object recognition

The other example was to recognize objects such as flower, wall, ground and wall edges. The aim of process was to do real-time image processing exercises by finding the object randomly placed on the ground by the mobile robot. There were also 21 input neurons in the input layer, 42 neurons in the hidden layer and 9 neurons in the output layer for this structure. The number of iteration was set to 1.500, the learning and momentum coefficients were set to $\alpha = 0.9$ and $\mu = 0.7$, respectively. The ANN structure was trained with 98 data sets and then tested with 17 unseen data sets. The errors in training and test were 8.4% and 11.1%, respectively. As expected, the correlation was lower ($r^2 = 0.51$) for this application.

5.3. Other applications

In order to show functionality of Web-SUN, three different tests (Test #1, Test #2 and Test #3) were also introduced in Sagiroglu et al. (2005, 2006), Yilmaz (2005), Yilmaz et al. (2006a, 2006b). Test #1 involves two different experiment sets. The first set deals with the recognition of the objects in different color and sizes. The other set was used to test the mobile robot if it passes the obstacle without any difficulty or crash. For this purpose, the position information of the boxes in Fig. 10a was estimated from the images acquired by the camera and the robot can move from one position to the target position smoothly. The mobile robot could also achieve any movement without touching the objects to reach the target correctly even if the locations of the objects were moved during the operations. During test processes, color-based image preprocessing, moment-based feature extraction and multilayered perceptron ANN were used for this experiment. Furthermore, pattern analysis based terrain characterization

(a) Test #1 (b) Test #2 (c) Test #3

Fig. 10. Other test scenarios for Web-SUN.

(visual sensing) for the obstacle avoidance step and 20×20 sized arrays with the calculated object and robot positions for mapping were performed.

In Test #2, the image data for flowers were stored to the platform at given time intervals to be used for different applications. For this purpose, the flowers were initially identified and the positions of each flower (the positions of the flowerpots) were determined. This test was shown in Fig. 10b while the mobile robot was in operation. The images of each plant (flower) were stored to the platform disk. The camera positions and time to take the pictures were determined by the operator. In the recognition algorithms used to estimate position and to make corrections, similar to Test #1, color-based image pre-processing, moment-based feature extraction and multilayered perceptron ANN for the last processes were used.

In the last test (Test #3), the line tracking capability of the mobile robot was examined. In order to achieve an observation route or a path was considered and painted on the ground with red line. During this test, Web-SUN was expected to follow the line with minimum error as illustrated in Fig. 10c. The real-time images of the line were captured by the camera and the position and orientation of the line was then estimated from processed images. According to this estimation, a control signal was achieved. It was then used as input to the position control algorithm to keep the robot in the route. From the experiments presented in this work, it was observed that the robot could follow the line or path accurately.

6. Results and conclusions

In this work, the mobile robot system Web-SUN especially designed for multi-functional tasks was successfully introduced to achieve real-time web-based exercises. The results have shown that the web platform presented in this work can be used for various tasks.

The results have also depicted that the real-time examples presented in this work might be used for some courses on real-time distance education, engineering applications, even for long life learning, image processing, robotics and programming.

Making use of the designed and developed control center, faster communication was achieved with TCP/IP protocol over web and UDP protocol over LAN. The autonomous structure of the robot reduces the intensity of communication between control center and operators. As a result, a number of low-cost real-time applications can be achieved.

The obtained test results show that web-based mobile robot platform can effectively support web-based multipurpose real-time experiments.

The significant contributions of this work are; presenting new concepts for web-based real-time applications, using only image processing for all activities on the web, supporting applications with its own library and requiring only web-browser on internet.

It is assumed that this web platform would help users (students and researchers) to do their practices, applications and homework using up-to-date technologies at their locations.

The authors are planning to use this platform for their engineering courses such as image processing, automatic control, algorithms, robotics, pattern recognition, programming and artificial intelligence. The platform might be also used in other real-time remote laboratory exercises and R&D activities to develop new real-time expert system and applications.

The difficulties faced in this work in realization of this web platform were the integration of various hardware and software components into one design environment, transferring images to the platform and processing them in short time, bandwidth limitation and controlling many components and peripherals remotely.

Acknowledgements

This project was partially supported by Science and Research Project Office of Selcuk University. The authors would like to thank Selcuk University for their financial support.

References

Guimaraes, E., Maffeis, A., Pereira, J., Russo, B., Cardozo, E., Bergerman, M., et al. (2005). REAL: A virtual laboratory for mobile robot experiments. *IEEE Transactions on Education, 46*(1), 37–42.

Halme, A., Leppanen, I., Suomela, J., Ylonen, S., & Kettunen, I. (2003). WorkPartner: Interactive human-like service robot for outdoor applications. *The International Journal of Robotics Research, 22*(7–8), 627–640.

Haykin, S. (1994). *Neural networks: A comprehensive foundation*. New York: Macmillan.

Hu, H. S., Yu, L. X., Tsui, P. W., & Zhou, Q. (2001). Internet-based robotic systems for teleoperation. *Assembly Automation, 21*(2), 143–151.

Jacobsen, S. C., Olivier, M., Smith, F. M., Knutti, D. F., Johnson, R. T., Colvin, G. E., et al. (2004). Research Robots for applications in artificial intelligence, teleoperation and entertainment. *The International Journal of Robotics Research, 23*(4–5), 319–330.

Kuc, R., Jackson, E. W., & Kuc, A. (2004). Teaching introductory autonomous robotics with JavaScript simulations and actual robots. *IEEE Transactions on Education, 47*(1), (74–82).

Kwon, Y., Rauniar, S., Chiou, R., & Sosa, H. (2006). Remote control of quality using Ethernet vision and web-enabled Robotic system. *Concurrent Engineering-Research and Applications, 14*(1), 35–42.

Maren, A., Harston, C., & Pap, R. (1990). *Handbook of neural computing applications*. London: Academic Press.

Marin, R., Sanz, P. J., & Del Pobil, A. P. (2003). The UJI online Robot: An education and training experience. *Autonomous Robots, 15*(3), 283–297.

Oboe, R. (2001). Web-interfaced, force-reflecting teleoperation systems. *IEEE Transactions on Industrial Electronics, 48*(6), 1257–1265.

Patel, S., Sanyal, R., & Sobh, T. (2006). RISCBOT: A WWW-enabled mobile surveillance and identification robot. *Journal of Intelligent & Robotic Systems, 45*(1), 15–30.

Pipe, A. G., & Carse, B. (2007). Fuzzy classifier system architecture for mobile Robotics: An experimental comparison. *International Journal of Intelligent System, 22*, 993–1019.

Robinette, M. F., & Manseur, R. (2001). ROBOT-DRAW, an internet-based visualization tool for robotics education. *IEEE Transactions on Education, 44*(1), 29–34.

Rosch, O. J., Schilling, K., & Roth, H. (2002). Haptic interfaces for the remote control of mobile Robots. *Control Engineering Practice, 10*(11), 1309–1313.

Rumelhart, D. E., & McClelland, J. L. (1986). *Parallel distributed processing*. Cambridge: The MIT Press.

Sagiroglu, S., Yilmaz, N., & Bayrak, M. (2005). Design and implementation of a web based mobile robot. In *virtual international conference on intelligent production machines and systems* (pp. 393–397).

Sagiroglu, S., Yilmaz, N., & Wani, M. A. (2006). Web robot learning powered by bluetooth communication system. In *ICMLA'06 5th International Conference on Machine Learning and Applications* (pp. 149–156).

Saucy, P., & Mondada, F. (2000). KhepOnTheWeb: Open access to a mobile robot on the internet. *IEEE Robotics & Automation Magazine, 7*(1), 41–47.

Schilling, K., Roth, H., & Splica, C. (2005). A tele-experiment on Rover motor control via internet. *Journal of Robotic Systems, 22*(3), 123–130.

Simmons, R., Fernandez, J. L., Goodwin, R., Koenig, S., & O'Sullivan, J. (2000). Lessons learned from Xavier. *IEEE Robotics & Automation Magazine, 7*(2), 33–39.

Smith, A. C., & Hashtrudi-Zaad, K. (2006). Smith predictor type control architectures for time delayed teleoperation. *The International Journal of Robotics Research, 25*(8), 797–818.

Stein, M. R. (2000). Interactive internet artistry. *IEEE Robotics & Automation Magazine, 7*(2), 28–32.

Trahanias, P., Burgard, W., Argyros, A., Hahnel, D., Baltzakis, H., Pfaff, P., et al. (2005). TOURBOT and WebFAIR. *IEEE Robotics & Automation Magazine, 12*(2), 77–89.

Yilmaz, N., (2005). Design, realization and applications of a web-based mobile robot system. PhD Thesis. Institute of Natural and Applied Sciences. Konya, Selcuk University (in Turkish).

Yilmaz, N., Sagiroglu, S., & Bayrak, M. (2006a). General aimed web based mobil Robot: SUNAR. *Journal of The Faculty of Engineering and Architecture of Gazi University, 21*(4), 745–752.

Yilmaz, N., Sagiroglu, S., & Bayrak, M. (2006b). Landmark detection via ANN for web-based autonomous mobile robot: SUNAR. *Selcuk Journal of Applied Mathematics, 7*(1), 45–62, SJAM.