Strojniški vestnik - Journal of Mechanical Engineering 55(2009)6, 356-368
UDC 007.52
Paper received: 22.01.2008
Paper accepted: 08.07.2009

Internet Based Teleoperation for Cooperative Navigation and Manipulation

Ivan Petrović[1,*] - Fetah Kolonić[1] - Peter Korondi[2]
[1] University of Zagreb, Faculty of Electrical Engineering and Computing, Croatia
[2] Budapest University of Technology and Economics, Department of Automation and Applied Informatics, Hungary

The Internet is a very fast evolving new technology, allowing people to electronically connect to places that are thousands of miles apart. By combining network technologies with the capabilities of mobile robots and manipulators, Internet users can discover and physically interact with far-away places thereby, creating opportunities of resource sharing, remote experimentation and long-distance learning.

This paper presents a general concept of Internet based teleoperation, which structures a teleoperation system into three layers. The concept was applied in developing a teleoperation system that enables two human operators to safely control two cooperative mobile robots in unknown and dynamic environments from any two PCs connected to the Internet by installing a developed client program on them and by using simple force feedback joysticks. On graphical user interfaces, the operators receive images forwarded by the cameras mounted on robots, and on the joysticks they can feel forces forwarded by developed obstacle prevention algorithm based on the dynamic window approach. To overcome the instability caused by the unknown and varying time delay an event-based teleoperation method is employed to synchronize actions of each robot with commands from its operator. Through experimental investigation it is confirmed that developed teleoperation system enables the operators to successfully accomplish cooperative navigation and manipulation tasks in complex environments.

Keywords: internet, teleoperation, cooperative navigation, manipulation

0 INTRODUCTION

Teleoperation is a process where the operator faces task at some remote hazardous or inaccessible environment like space, underwater, nuclear plants, where they cannot physically be present. A teleoperator system extends the operator's capability to be able to work at the Remote Workplace. In about 1945 the first modern master-slave system teleoperator was developed by Goertz at Argonne National Laboratory (ANL) [1]. This system, which was a mechanical pantograph mechanic system, allowed a human operator to manipulate radioactive materials in a "hot cell" from outside. By using a master handle, the human operator could move the slave tong located inside the hot cell and receive force reflection. About ten years later (1954) electrical servomechanisms replaced direct mechanical linkages between master and slave devices [2]. Closed circuit television was introduced so that the operator could stay at an arbitrary distant place. In addition, the contact force between the slave and its environment was returned to the master arm. In 1964, Mosher developed an impressive piece of work, which was a handy-man containing electrohydraulic arms with ten degrees of freedom each. The problem of remote control of robotic systems has been the subject of much research in recent years. Remote control of robotic systems has been applied in manufacturing, underwater manipulation, storage tank inspection, nuclear power plant maintenance, space exploration, etc.

With a rapid development of information technology, the Internet has evolved from a simple data sharing media to an amazing information world where people can enjoy various services, teleoperation being one of them. The use of Internet for teleoperation tasks has become one of the most topical subjects in robotics and automation, see e.g. [3] to [6]. On the other hand, the Internet also entails a number of limitations and difficulties, such as restricted bandwidth, arbitrarily large transmission delays, delay jitter, and packet lost or error, all of which influence the performance of Internet based telerobotics systems. A number of approaches

[*]Corr. Author's Address: University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, HR-10000 Zagreb, Croatia, ivan.petrovic@fer.hr

have been proposed to ensure stability of the force feedback loop closed over the Internet, with the majority of them being based on passivity theory [3], [7] and [8] or on the event based action synchronization using a non-time reference [4], [9] and [11]. The latter approach is used in this paper because of its simplicity and effectiveness.

Conventional teleoperated robots rely on visual contact with the operator, either directly or through video transmissions. This is feasible only if the surroundings of the robot are known and static. However, teleoperation is often employed in controlling robots navigating and manipulating in unknown and dynamic environments. Guiding such a robot only with visual feedback is a formidable task, often complicated by the limited view from the camera. Under such conditions, a human teleoperator must exercise extreme care, especially in obstacle-cluttered environments. In order to increase the system performance and to reduce the operator stress and task errors, force feedback from the robot to the human operator is usually employed, see e.g. [12] and [13].

There are many complicated and sophisticated tasks that cannot be performed efficiently by a single robot or operator, but require the cooperation of a number of them. The cooperation of multiple robots and/or operators is particularly beneficial in cases when robots must operate in unknown and dynamic environments. Teleoperation of multiple robots by multiple operators over the Internet has been extensively studied for couple of years. A good survey can be found in [14].

In this paper a general concept of the Internet based teleoperation is presented and applied in designing a teleoperation system that consists of two cooperative mobile robots. In order to guarantee safe robot navigation in a dynamic environment we have developed a new obstacle avoidance algorithm based on the dynamic window (DW) approach introduced in [15]. The main advantage of our algorithm is that it takes robot dynamic constraints directly into account, which is particularly beneficial for safe navigation at high-speeds as the safety margin depends not only on distances between the robot and the nearby obstacles, but also on the velocities of their motion. The algorithm is implemented on both robots and each robot considers the other one as the moving obstacle. The two robots cooperate in doing the manipulation tasks, where one of them does the manipulation and the other serves as active vision system providing the operators with the third dimension in visual feedback.

The paper is structured as follows. In Section 1, we present a general concept of teleoperation based on layered structure with three layers on both master and slave sites. The developed teleoperation system implemented according to the general concept is described in Sections 2. Section 3 describes the experiments conducted in order to verify the developed system as well as the experimental results. Finally, concluding remarks are given in Section 4.

1 GENERAL CONCEPT OF THE INTERNET BASED TELEOPERATION

In the early days of teleoperation, the hands of a human operator and the remote environment were mechanically coupled. Therefore, the information was transferred mechanically which limited the type of the tasks that they could perform. Nowadays, the information is transferred electronically via communication systems, which opened a new dimension at the slave side. The internet enables a widespread use of teleoperation and wireless communication enables slave devices to move through remote environment. Therefore, today's teleoperation systems are human-robot systems, where a human and a robot simultaneously work together to achieve a task. In general, the slave device can be a multi-robot system as well, where two or more robots work together to achieve a task. Computer networked control enables systems to be distributed. The concept of an operator and manipulator can be replaced with some combination of multiple operators, multiple robots, and maybe even multiple assistant agents that work together in a cooperative environment [16].

An Internet-based teleoperation system is generally realized as a bilateral system consisting of master and slave devices which interact through a bidirectional communication channel, as shown in Fig. 1. This interaction strongly couples two processes. One process is the interaction between the operator and the master device, the other is the interaction between the slave device and the remote environment. The

master device represents the distant environment at the operator site, and the slave device represents the operator at the remote site. When humans use this device only to move objects, the reaction force from the distant environment has no significant effect on the performance, so measurement of the operator's position and visual feedback may be enough. However, if such tasks are to be performed when the reaction of the environment is important and the slave device can do damage in the remote environment, for example when moving through it, screwing a bolt or assembling something, then a force feedback is needed to improve efficiency. Force feedback increases the feeling of being there.

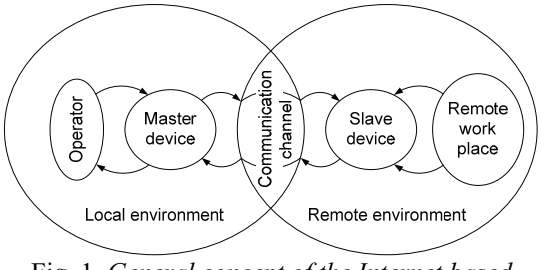

Fig. 1. *General concept of the Internet based teleoperation*

1.1 Functional Structure of Teleoperation Systems

Teleoperation systems can be physically realised in many different ways, but functionally, at the abstract level, they are generally organised in three main layers as shown in Fig. 2. The layers can communicate vertically with the local layers as well as horizontally with the remote layers via the Internet. The Sensor-Monitor Functional Layer provides a human operator with information about a remote environment, which is needed for effective teleoperation. The Sensor Layer at the Slave Site contains necessary, usually non-contact, sensors to observe the remote environment and the Monitor Layer at the Master Site shows its processed data, usually at non-contact displays or indicators. The Manipulator Functional Layer makes the effective work in the remote workplace. This layer typically involves a slave device and may also contain contact sensors. At the Master Site, the Manipulator Layer has a real contact with the Operator. This layer typically involves the master device. This layer, included with the master device, may contain contact sensors, and contact actuators. The contact actuators relay the sensed data of the contact sensors of the Manipulator Layer. The Transporter Functional Layer at the lave Site includes a Transporter (mobile unit), which can carry the slave device (manipulator), and the Transporter Control Layer at the Master Site, which controls the Transporter at the Slave Site. This layer may contain services that can help to the navigation of the Transporter Layer at the Slave Site.

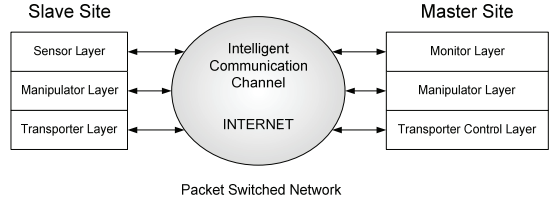

Fig. 2. *Functional layers definition for the general concept of the Internet based teleoperation*

1.2 Functional Layers Definition

The configuration of the Sensor-Monitor Functional Layer can be seen in Fig. 3. The non-contact sensors on the Slave Site are observing the Far Environment. The Intelligent Sensed Data Computing System at the Slave Site is a pre-processor for the sensed data. The purpose of this device is filtering, correction and compression. The audio and video compression formats are widely used on the Internet, to decrease the necessary bandwidth. The Computing System may contain a pattern recognition module to improve the efficiency and intelligence of the Slave Site. The purpose of the Intelligent Sensed Data Computing System at the Master Site is data decompression and lost data recovery.

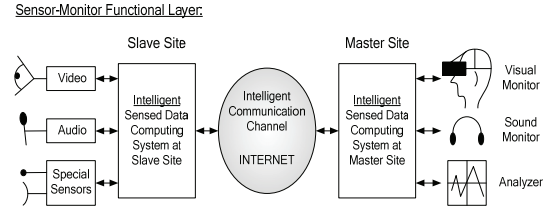

Fig. 3. *Sensor-Monitor Functional Layer definition for the general concept of the Internet based teleoperation*

The configuration of the Manipulator Functional Layer can be seen in Fig. 4. Typically, this layer includes the Slave and the Master devices, as mentioned above. The main purpose of this layer is the position control and the force feedback. The force feedback is a typical action-reaction process, representing information flow between the layers and between the Master and the Slave Site. The exact realization of the force feedback depends on the specific approach. The configuration of the Master Device is not equal to the configuration of the Slave Device in a general case. The main purpose of the Intelligent Trajectory Planning is to couple the master and the slave in position and force.

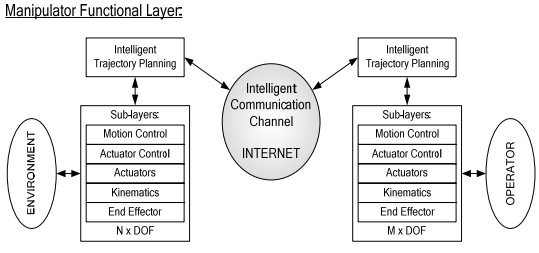

Fig. 4. *Manipulator Functional Layer definition for the general concept of the Internet based teleoperation*

The configuration of the Transporter Functional Layer can be seen in Fig. 5. The configuration of the Slave Site of this layer is very similar to Slave Site of the Manipulator layer.

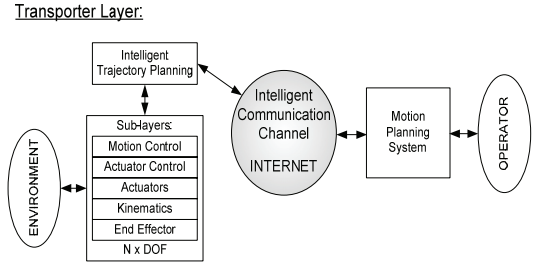

Fig. 5. *Transporter Functional Layer definition for the general concept of the Internet based teleoperation*

The Transporter Functional Layer has a connection with the Far Environment and carries the Slave Device and the necessary sensors. The Motion Planning System converts the Operator commands to the control commands for the transporter on the Slave Side and proceeds the feedback force from the Slave Side to the Operator. This device may use various advanced services for navigation.

2 A CASE STUDY OF THE INTERNET BASED TELEOPERATION SYSTEM

2.1 Physical Embodiment of the System

The teleoperation system considered in this paper is schematically illustrated in Fig. 6. It consists of two cooperative mobile robots that operate in a remote environment (one of them serves as *Scout* and the other one as *Mobile Manipulator*) and two operator stations with PCs and force feedback joysticks. Scout explores the remote site and with its sensory information assists the operator controlling the Mobile Manipulator, which executes tasks of direct interaction with the working environment by using the robot arm mounted on it. While the operators' PCs (clients) are directly connected to the Internet, robots' on-board PCs (servers) are connected to Internet via the wireless LAN. Each operator controls a single robot and receives visual and other data from the robot controlled by the other operator. Observers can connect to one or both robots and receive sensory data from them, but with no control over robots actions.

Used mobile robots are *PIONEER 2DX* (Scout) and *PIONEER 3DX* with *Pioneer arm* with five degrees of freedom and a gripper mounted on it (Mobile Manipulator), all manufactured by *MobileRobots Inc.* Scout is equipped with an on-board PC, a ring with sixteen sonar sensors, laser distance sensor and a *Sony EVI-D30* Pan-Tilt-Zoom (PTZ) camera. Mobile Manipulator carries an external laptop computer, a ring with sixteen sonars and a *Cannon VC-C50i* PTZ camera.

An operator station can consist of any personal computer with an adequate Internet connection, a force feedback joystick and developed client application. The used joystick is the *Logitech WingMan Force 3D* Joystick, which has two axes on which it can read inputs and generate force: x (stick up-down) and y (stick left-right) and two additional axes that can only read inputs: z (stick twist) and throttle.

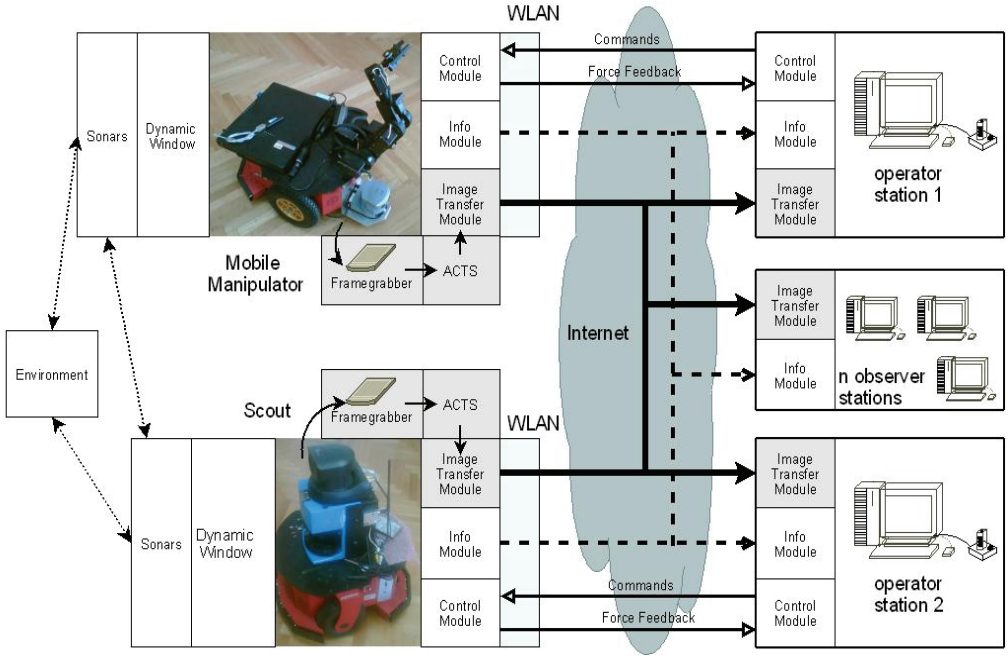

Fig. 6. *Scheme of the physical embodiment of our internet based teleoperation system*

Developed client application with its Graphical User Interface (GUI), shown in Fig. 7, enables the operator to continuously supervise both mobile robot actions but also to establish communication connection between master and slave sides, preset connection type and system operation mode.

Communication between server applications running on the mobile robots' on-board PCs and client applications running on the operators' PCs is initialized and terminated by client applications. In special cases, e.g. when a robot is turned off or malfunctioning or in case of communication failure, a server application can refuse or terminate the connection. Communication is established by using three independent communication modules: control module, image transfer module and info module. Modules are executed within separate threads of applications and communicate by using separate communication sockets. Modular structure decreases individual socket load and enables each module to transfer specific type of data without employing complicated packet scheduling schemes. Control module is used to transmit commands from the joystick to the robot and force feedback signal from the robot to the joystick (Control messages field on GUI). Image transfer module transmits the frames of visual feedback signals from robots' cameras to operators via GUI of the client application (Control/Observation field on GUI). Info module transmits time non-critical information such as robot velocities, arm joint angles and camera pose (Info field on GUI).

By presetting the connection type (active or passive) and the operation mode, the operator actually activates corresponding functional layer(s) of the teleoperation system. For example, if passive connection is selected the station becomes an observer station and not an operator station. Only basic functions of the Sensor-Monitor Functional Layer are activated, enabling the observer to supervise robots activities via visual feedback. If active connection is selected, the station becomes a real operator station enabling the operator to teleoperate the robot with which the connection is established. In active connection type, the operator can choose one of the three operation modes (driving, manipulation and observation modes), which activate corresponding teleoperation layers as described below.

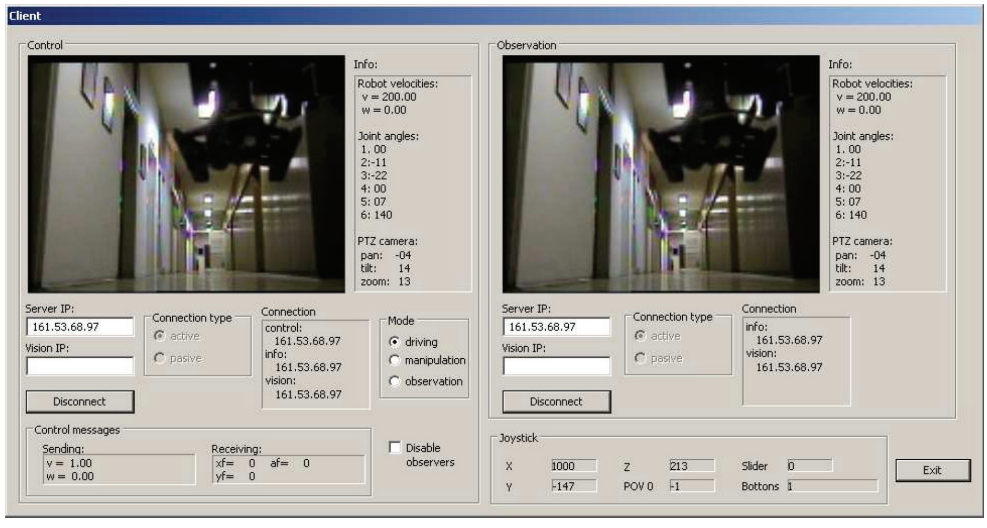

Fig. 7. *Graphical User Interface at Operators' Stations*

2.1 Implementations of the Functional Layers

2.2.1 Sensor-Monitor Functional Layer

The Sensor-Monitor Functional Layer is implemented according to the configuration shown in Fig. 3, but audio signal is not yet implemented. The operator receives the frame streams of visual feedback signals from both robots' cameras via GUI. Our system makes it possible for operators to receive visual feedbacks from both mobile robots so they can view the working environment from two different perspectives. In this way, operators can have a better sense about the third dimension, information not normally present in a single visual feedback. There are a number of difficulties in implementing a system with visual feedback [17]. Sensors and image acquisition hardware are expensive and directly affect visual feedback quality. High-quality visual feedback means that more information needs to be transferred in less time and for that more bandwidth or image compression is necessary. Also, it is not simple to transfer visual signal to more clients because limited processing power of the mobile robot needs to be shared among a number of tasks.

All these issues result in visual feedback delay, limited image acquisition frequency and lower video quality. Because human operator and visual feedback form part of the same control loop (human-in-the-loop, HITL) it is important to keep delays below 100 [ms] in order to keep the system interactive [18]. Frame rate should not be lower than 10 frames per second so that the operator does not notice individual frames, but fluent video display [18]. In order to fulfill this requirement we use *ActivMedia Color Tracking System* (ACTS) to fetch the images from the cameras connected to the PCs via the frame grabber cards. ACTS is an application which, in combination with a colour camera and frame grabber hardware in a PC, enables custom applications to actively track up to 320 colour objects at a full 30 [fps] image acquisition [19]. ACTS originally serves images to applications connected to it through TCP/IP, but that functionality is not used because it cannot control frequency and order in which clients receive images. Therefore, we have developed a new communication module, whose server side periodically requests an image form the ACTS and sends it to the connected clients.

Since robots are mobile and operate in an unknown environment, quality of visual feedback is crucial both for successful task execution and for safety reasons. A possibility is given to the operator to observe the robot surrounding environment and adjust the pan, tilt and zoom of the camera at the beginning of a teleoperation session by selecting the *observation mode* of operation and by moving the joystick. The joystick x axis is used to control tilt angular velocity of the camera, y axis to control its pan angular velocity, throttle for adjusting zoom

value, and one joystick button sends the camera to its home position. The operator can track the changes of pan, tilt and zoom values at Info field of the GUI.

2.2.2 Manipulator Functional Layer

Manipulator Functional Layer is implemented only on the Mobile Manipulator robot. It is activated when the *manipulation operating mode* is chosen at GUI. The arm is controlled joint by joint, where two joystick buttons are used to choose one of the arm's 6 joints, third button sends the arm in its home position and *y* axis is used to input the chosen joint velocity and to display the reflected force.

During robot arm movement force is reflected when the joint being controlled rotates to an angle that is less than 10° away from its maximum value

$$F_{arm} = F_{\max}\left(1-\frac{\xi_{\max}-\xi_i}{10}\right),\ \left|\xi_{\max}-\xi_i\right|<10°, \qquad (1)$$

where F_{arm} is the reflected force amplitude corresponding to the current joint angle ζ_i, F_{max} maximal reflected force amplitude, and ζ_{max} maximal joint angle. Feedback force informs the Manipulator's operator that the controlled joint approaches its maximal angle.

2.2.3 Transporter Functional Layer

The Transporter Functional Layer is activated when the client application is actively connected to the mobile robot and *driving operating mode* is chosen at the GUI. When this layer is active the joystick *x* and *y* axes are used to input the desired translational and rotational velocities, respectively. Force feedback is applied on the same two axis, defying commands that would lead the robot toward detected obstacles. In order to guarantee safe robots navigation in dynamic environments and/or at high speeds, it is desirable to provide a sensor-based collision avoidance scheme on-board the robots. Due to this competence, the robots can react without delay on changes in its surrounding. The usually used methods for obstacle prevention are based on virtual forces creation between the robot and the closest obstacle, see e.g. [20] and [21], where the force is inversely proportional to the distance between them. We have developed a new obstacle avoidance algorithm based on the dynamic window (DW) approach [15]. It takes robot dynamic constraints directly into account, which is crucial for safe navigation at high-speeds as the safety margin depends not only on distances between the robot and the nearby obstacles, but also on the velocity of robot motion. The algorithm is implemented on both robots and each robot considers the other one as the moving obstacle.

The DW produces trajectories that consist of circular and straight line arcs. It is usually integrated with a global path planning algorithm, e.g. FD* algorithm as in [22] for executing autonomous tasks in partially unknown environments. While global path planning algorithm calculates optimal path to a specific goal, the DW algorithm takes into account the unknown and changing characteristics of the environment based on the local sensory information. We use the DW algorithm in a teleoperation system, without a global path planning algorithm, just to ensure safe motion of the mobile robot and to help the operator to better perceive obstacles. Therefore, some modifications to the original algorithm have to be made.

Operator issues translational and rotational velocities' references and the DW algorithm evaluates the given commands while taking into account local sonar readings and kinematic and dynamic robot constrains. Commands that are not safe are not executed and force feedback is generated in order to warn the operator. The proposed DW-based collision prevention algorithm consists of the following steps:

1. Desired velocities (v_d, ω_d) are fetched from the buffer and constrained by maximal and minimal achievable velocities,
$$v_d \in \left[v_{\min}, v_{\max}\right],\ \omega_d \in \left[\omega_{\min}, \omega_{\max}\right]. \qquad (2)$$
2. Resulting velocities are additionally constrained to values from the set of velocities $V_{nc} = \{v_{nc}, \omega_{nc}\}$ achievable in one robot control cycle
$$v_d \in v_{nc} = \left[v_c - T\,\dot{v}_m, v_c + T\,\dot{v}_m\right],$$
$$\omega_d \in \omega_{nc} = \left[\omega_c - T\,\dot{\omega}_m, \omega_c + T\,\dot{\omega}_m\right], \qquad (3)$$
where v_c and ω_c are current velocities, v_m and ω_m are maximal accelerations/decelerations and T is robot control cycle duration.
3. Minimal stopping path is calculated. Stopping time and applied translational deceleration must be established first. If condition

$$\left|\frac{v_d}{\dot{v}_m}\right| > \left|\frac{\omega_d}{\dot{\omega}_m}\right| \quad (4)$$

is satisfied, maximal translational deceleration a_s is applied during stopping time

$$t_s = \left|\frac{v_d}{\dot{v}_m}\right|,\ a_s = \dot{v}_m. \quad (5)$$

If (4) is not satisfied, it takes more time to stop the rotation of the robot than its translation. Then translational deceleration

$$t_s = \left|\frac{\omega_d}{\dot{\omega}_m}\right|,\ a_s = \left|\frac{v_d}{t_s}\right|, \quad (6)$$

smaller than maximal is applied. Minimal stopping path s_s is then

$$s_s = v_d\left(t_s + t_{stm}\right) - \frac{a_s t_s^2}{2}, \quad (7)$$

where t_{stm} is additional safety time margin.

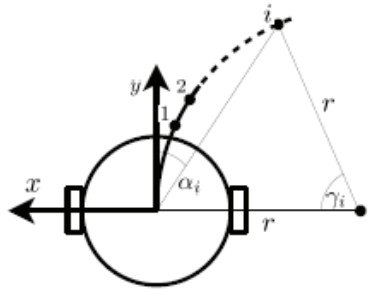

Fig. 8. *A robot trajectory described as a circular arc*

4. For the given velocities (v_d, ω_d) coordinates of N_p points on circular or straight line arc (Fig. 8) are calculated using following equations

$$\gamma_i = 2\alpha_i = 2T N_c \frac{i}{N_p},$$
$$x_i = x_c + \mathrm{sgn}(v_d) r \sin(\gamma_i),$$
$$y_i = y_c + \mathrm{sgn}(v_d)\mathrm{sgn}(\omega_d) r \cos(\gamma_i), \quad (8)$$

where (x_c, y_c) is the center of the arc, $r = v_d/\omega_d$ its radius, i = 1,.., N_p is the index specifying the point on the trajectory, from 1 to N_p, and N_c is the number of cycles for which the algorithm is executed.

5. Minimal allowed distance to obstacle ρ_{min} is

$$\rho_{\min} = r_r + s_{c1} + (s_{c2} - s_{c1})\frac{v_d}{v_{\max}} + s_{c\omega}\frac{\omega}{\omega_{\max}}, \quad (9)$$

where r_r is the robot radius, s_{c1} safety clearance at low translational velocities, s_{c2} safety clearance at high translational velocities and $s_{c\omega}$ rotational safety clearance. Rotational safety clearance is set to $s_{c\omega} = 0$ because it is considered safe to rotate the robot in place, i. e. without any translation. This clearance can be set to a higher value if rotation in place is not considered safe, e.g. when rotating the robot with extended robot arm. At low translational velocities (i.e. velocities close to zero) s_{c2} contributes little or not at all to ρ_{min} ($\rho_{min} = r_r + s_{c1}$, $v_d = 0$, $s_{c\omega} = 0$). At high translational velocities (i.e. velocities close to maximum value) s_{c1} has little or no influence on ρ_{min} value ($\rho_{min} = r_r + s_{c2}$, $v_d = v_{max}$, $s_{c\omega} = 0$).

6. For every point on the trajectory (v_d, ω_d), starting with the one closest to the robot, distance to the closest obstacle $\rho_i(v_d, \omega_d)$ is calculated and if the condition

$$\rho_i(v_d, \omega_d) \geq \rho_{\min}(v_d, \omega_d) \quad (10)$$

is true, the calculation is executed for the next point on the trajectory. If (10) is satisfied for all N_p points on the trajectory then it is considered clear, force feedback is zero and the algorithm is finished.

7. If the condition (10) is not satisfied, path to the *i*-th point on the trajectory is calculated as:

$$s_p = v_d \frac{i}{N_p} T N_c. \quad (11)$$

8. If the stopping path s_s is shorter than the path to the i-th point s_p

$$s_s < s_p, \quad (12)$$

i.e. it is possible to stop the robot before it comes closer than ρ_{min} to the obstacle, trajectory is considered safe, force feedback is calculated and the algorithm is finished. Force feedback is calculated as follows:

$$F_{amp} = \beta\sqrt{x_o^2 + y_o^2},\ F_{ang} = a\tan 2(y_o, x_o), \quad (13)$$

where F_{amp} and F_{ang} is force feedback amplitude (scaled to [0, 1] with scaling factor β) and direction, respectively, (x_o, y_o) is the closest obstacle position in mobile robots coordinates and *atan2* is arctangent function.

9. If the condition (12) is not met, the investigated trajectory leads to collision, i.e. leads robot closer than ρ_{min} to the obstacle. To prevent collision, desired velocities magnitudes are decreased as follows:

$$v_{d(i+1)} = v_{di} - v_{d1}/N_s ,$$
$$\omega_{d(i+1)} = \omega_{di} - \omega_{d1}/N_s , \quad (14)$$

where $v_{d(i+1)}$ and $\omega_{d(i+1)}$ are velocities for the next iteration of the algorithm, v_{di} and ω_{di} are velocities of the current iteration, v_{d1} and ω_{d1} are original commanded velocities that entered the first iteration of the algorithm and N_s is the number of steps in which velocities are decremented. If the new values are different than zero, algorithm is repeated from the 3rd step until safe trajectory is found.

2.3 Event Based Action Synchronization

Force feedback loop between Slave and Master Site is established in cases when the Manipulator Functional Layer or Transporter Functional Layer is active. While in the former case the feedback force warns the Operator that the controlled joint approaches its maximal angle, in the latter case it warns the Operator that the moving robots approaches close to an obstacle. The main drawback of closing a feedback loop over the Internet is the existence of stochastic and unbounded communication delay that can affect system performance and make the force feedback loops unstable.

These problems are usually addressed by ensuring passiveness of the feedback loop, see e.g. [3], [7] and [8] or by using a non-time reference for action synchronization, as presented in [4], [9] and [10]. The latter approach is used here, because there is no need for additional signal processing and consequently, the control system is computationally much simpler. The stability of the control system with event-based action synchronization is ensured if a non-decreasing function of time is used as the action reference [23]. The number of completed control cycles, which is obviously a monotone increasing function of time, was chosen for this reference. Each control cycle (Fig. 9) is a sequence of the following actions:

1. Server application fetches the most recent force feedback from the buffer and sends it to the client application.
2. Client application receives force feedback.
3. Received force is applied to the joystick.
4. New commands are read from the joystick.
5. The commands are sent to the server application.
6. Server application receives new commands and refreshes the buffer. Depending on the operation mode, the DW algorithm or arm controller periodically fetches the command from the buffer and refreshes the force feedback on the buffer after its execution. Commands are refreshed once for every force feedback signal sent to the client application.

The proper order of arrival of information packets is crucial for stability of the control system, even more than their timely delivery. For this reason, User Datagram Protocol (UDP) is used for sending operator commands and force feedback. UDP, unlike TCP, does not resend lost packets and is therefore, more suitable for real time applications. Additionally, UDP packets tend to be smaller in size than corresponding TCP packets which yields a smaller load on the communication channel. However, unreliable delivery of control packages could break the control cycle and longer communication delays may destabilize the system. To avoid this, server application monitors control cycle duration and if it exceeds a prescribed maximal value (e.g. 1 s), server application initiates a new control cycle by sending a fresh force feedback packet. All packets that arrive outside their control cycles are simply ignored.

The described event-triggered control ensures that the force applied to the joystick corresponds to the command sent in the previous control cycle and that the buffer state is refreshed with the command that corresponds to the force feedback sent in the current control cycle. Therefore, action synchronization is achieved using the control cycle number as a time independent reference and the system is stable. Action synchronization between Mobile Manipulator and its operator station is executed independently from action synchronization between Scout and its operator station. This arrangement is referred to as decentralized event-based control [14] and it assures that control cycle

duration of one robot-operator station pair is not affected by communication delays of the other pair.

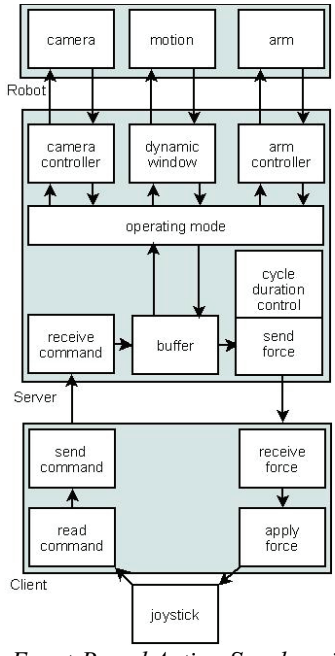

Fig. 9. *Event-Based Action Synchronization between client and server control modules*

3 EXPERIMENTAL RESULTS

In order to validate the developed system, a number of experiments have been carried out and different signals have been recorded. The results of four illustrative experiments are given here.

In the first experiment the operator drove the robot around an obstacle. The commanded velocities on client side (CS), commanded velocities on server side (SS), measured velocities, applied force feedback direction and amplitude were recorded. During the first phase of the experiment (from 0 to 3 s, Fig. 10), the robot moved forward and the obstacle was outside the DW sensitivity region. Measured velocities matched the commanded ones and no force was reflected. When the operator started turning the robot, the obstacle entered DW sensitivity region and force was reflected at −80° informing the operator that the obstacle was on the right from the robot (from 3 to 6 s), and at the same time DW limited the robot angular velocity, while translational velocity matched the commanded one. At approximately t = 6 s, the operator stopped turning the robot, and the reflected force fell down, DW algorithm limited for a short time, first the robot translational velocity and then angular velocity. Then the robot moved away from the obstacle and both velocities matched the commanded ones, but the reflected force again slowly increased, which was caused by the next obstacle entering DW sensitivity region.

In the second experiment two operators, located at different places, attempted to drive the Scout and Mobile Manipulator robots into a head-on collision. This is the most difficult case for the modified DW algorithm to handle as both robots see the other as a moving obstacle.

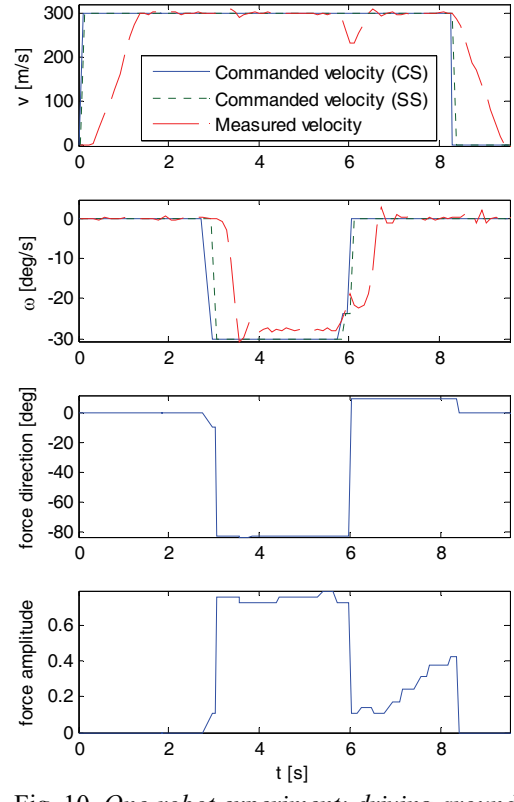

Fig. 10. *One-robot experiment: driving around a corner*

Velocities of both robots dropped rapidly when the distance between them approached the DW sensitivity range (at approximately t = 3 s, Fig. 11). At the same time reflected forces sent to both operators rose.

When force feedback signals reached their maximal values velocities of both robots dropped to zero. Robots stopped and the collision was avoided. Fluctuation of the Scouts's rotational

and translational velocities was due to the difficulties that its low-level velocity control system had while following reference values.

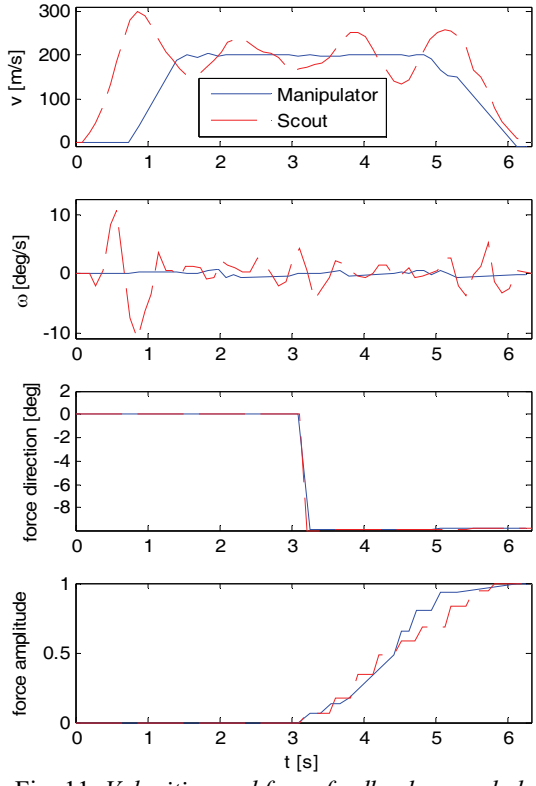

Fig. 11. *Velocities and force feedback recorded during the two-robot collision experiment*

In spite of this, the modified DW algorithm successfully prevented robots from colliding. Force feedback angle should be 0° during this experiment as the obstacle is directly in front of the robot. However, this angle varies between 0 and 10°. No stochastic processing was implemented and sonars were treated as rays whose orientation depends on the sonar's position on the robot. Such an error is tolerated due to the fact that force feedback is used only to provide the operator with a general notion about the position and distance to the closest obstacle. In the last two experiments the task was to pick up a small cardboard box off the floor, place it on the Mobile Manipulator's back and put the arm at the home position. Only one operator controlling the Mobile Manipulator accomplished the task after few attempts and with some unnecessary adjustments (Fig. 12). Adjustments were needed as the camera is mounted close to the ground (for better object grasping) not covering the entire workspace of the arm. Another problem is the lack of information about the third dimension, which complicates the adjustments of joint angles even when the arm is visible to the operator. The same task can be accomplished much easier and faster (approximately 60 seconds in contrast to approximately 110 seconds) if Scout and Mobile Manipulator cooperate (Fig.13).

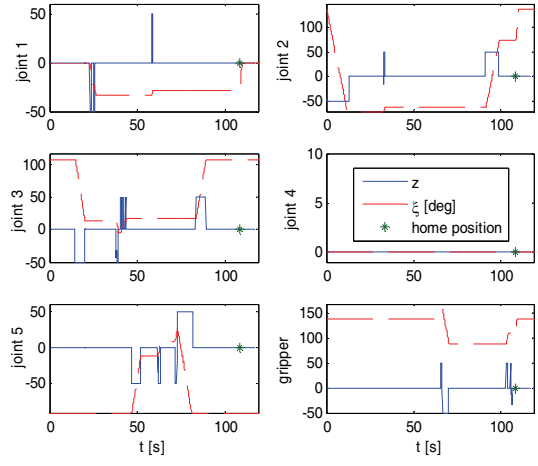

Fig. 12. *Motion of the robot arm joint angles during the one-robot experiment*

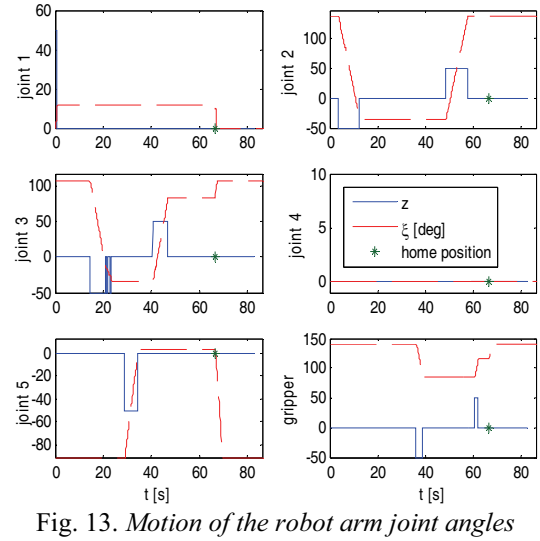

Fig. 13. *Motion of the robot arm joint angles during the two-robots experiment*

Scout's assistance gives the Mobile Manipulator's operator a better view of the distant site enabling him to see the arm during the entire procedure and giving him a feel of the third dimension.

4 CONCLUSION

A teleoperation system that enables two human operators to safely control two cooperative mobile robots in unknown and dynamic environments over the Internet has been developed. Each operator receives images displayed on the graphical user interface, which are forwarded by the cameras mounted on the robots, and force feedback on the joystick, which is reflected from the robot controlled by them. To overcome the instability caused by the unknown and varying time delay, event-based teleoperation system is employed to synchronize actions of each robot with command from its operator. The teleoperation system has been implemented according to the general teleoperation concept, which is based on a layered structure with three main layers on both master and slave sites. Through experimental investigation it has been confirmed that developed teleoperation enables the operators to successfully accomplish teleoperation tasks in complex environments. The developed teleoperation system could be easily adjusted to different robot and sensor types to allow application in many various tasks.

5 REFERENCES

[1] Sheridan, T.B. (1989) Telerobotics, *Automatica* 25(4), p. 487-507.

[2] Goertz, R.C., Thompson, W.C. (1954) Electronically controlled manipulator, *NUCLEONICS,* 12 (11), pp. 46-47.

[3] Munir, S. (2001) Internet-Based Teleoperation, *PhD thesis*, Georgia Institute of Technology, Atlanta.

[4] Wang, M., Liu, J.N.K. (2005) Interactive control for Internet-based mobile robot teleoperation, Robotics and Autonomous Systems, 52, p. 160-179.

[5] Elhajj, I., Xi, N., Liu, Y.H. (2000) Real-time control of the Internet based teleoperation with force reflection, *Proc. IEEE Int. Conf. Robotics and Automation*, San Francisco, CA, April 24-28, p. 3284-3289.

[6] Lo, W., Liu, Y., Elhajj, I.H., Xi, N., Wang, Y., Fukada, T. (2004) Cooperative teleoperation of a multirobot system with force reflection via Internet, *IEEE/ASME Transactions on Mechatronics*, 9(4), p. 661-670.

[7] Niemeyer, G. (1996) Using wave variables in time delayed force reflecting teleoperation, *PhD thesis*, Massachusetts Institute of Technology, Cambridge, MA.

[8] Niemeyer, G., Slotine, J.J.E. (2004) Telemanipulation with time delays, *The International Journal of Robotics Research*, 23(9), p. 873-890.

[9] Brady, K., Tarn, T.J., (2002) Handling latency in Internet-based teleoperation, (K. Goldberg, R. Siegwart (Eds.)), Beyond Webcams: An Introduction to Online Robots, *The MIT Press, London*, ISBN 0-340-80656-7, p. 171-192

[10] Luo, R.C., Su, K.L. (2003) Networked intelligent robots through the Internet: issues and opportunities, *IEEE Proc*, 91(3), p. 371-382.

[11] Petrović, I., Babić, J., Budušić, M. (2007) Teleoperation of collaborative mobile robots with force feedback over Internet, *Proceedings of the Fourth International Conference on Informatics in Control, Automation and Robotics*, Angers, France, May 9-12, p. 430-437.

[12] Sheridan, T.B. (1992) Telerobotic, Automation, and Human Supervisory Control, *The MIT Press*, Cambridge, ISBN 0-262-19316-7, MA, USA.

[13] Lee, S., Sukhatme, G., Kim, G., Park, C. (2002) Haptic Control of a Mobile Robot: A user study, *IEEE/RSJ International Conference on Intelligent Robots and Systems EPFL*, Lausanne, Switzerland, Sept. 30 – Oct. 4, p. 2867–2874.

[14] Lo, W., Liu, Y., Elhajj, I.H., Xi, N., Wang, Y., Fukada, T. (2004) Cooperative teleoperation of a multirobot system with force reflection via Internet, *IEEE/ASME Transactions on Mechatronics*, 9(4), p. 661-670.

[15] Fox, D., Burgard, W., Thurm, S. (1997) The Dynamic window approach to collision avoidance, *IEEE Robotics & Automation Magazine*,4(1), p.23-33.

[16] Dalton, B., Taylor, K. (1998) A framework for Internet robotic*, Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, WS2, p. 15-22.

[17] Herakovič, N. Computer and machine vision in robot-based assembly, *Strojniški vestnik* -

Journal of Mechanical Engineering 53 (2007)12, p. 858-873.

[18] Shaw, C., Green, M., Lang, J., Sun, Y. (1993) Decoupled simulation in virtual reality with the MR toolkit, *ACM Transactions on Information Systems*, 11(3), p. 287-317.

[19] ActivMedia, (2003) ACTS ActivMedia Robotics ColorTrack ing System - User Manual*, ActivMedia Robotics LLC*, Amherst.

[20] Borenstein, J., Koren, Y. (1990) Teleautonomous guidance for mobile robots. *In IEEE Transactions on Systems, Man and Cybernetics*, 6(20), p. 1437-1456.

[21] Lee, D., Martinez-Palafox, O., Spong, M.W. (2006) Bilateral teleoperation of a wheeled mobile robot over delayed communication network, *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*. Orlando, Florida, May 15-19, p. 3298 – 3303.

[22] Seder, M., Maček, K., Petrović, I. (2005) An integrated approach to real-time mobilerobot control in partially known indoor environments*, Proceedings of the 31st Annual Conference of the IEEE Industrial Electronics Society*, p. 1785-1790, Raleigh, North Carolina.

[23] Xi, N., Tarn, T.J. (2000) Stability analysis of nontime referenced internet based telerobotic systems, *Robotics and Autonomous Systems*, 32, p. 173-178.