Architecture of a Cyberphysical Avatar

Song Han[*], Aloysius K. Mok[*], Jianyong Meng[*], Yi-Hung Wei[*], Pei-Chi Huang[*], Xiuming Zhu[*], Luis Sentis[†]
Kwan Suk Kim[†], Risto Miikkulainen[*], Jacob Menashe[*]

[*]Department of Computer Science, The University of Texas at Austin
{*shan, mok, jmeng, yhwei, peggy, xmzhu, risto, jmenashe*}*@cs.utexas.edu*
[†]Department of Mechanical Engineering, The University of Texas at Austin
{*lsentis@austin.utexas.edu, kskim@utexas.edu*}

Abstract— **This paper introduces the concept of a cyberphysical avatar which is defined to be a semi-autonomous robotic system that adjusts to an unstructured environment and performs physical tasks subject to critical timing constraints while under human supervision. Cyberphysical avatar integrates the recent advance in three technologies: body-compliant control in robotics, neuroevolution in machine learning and QoS guarantees in real-time communication. Body-compliant control is essential for operator safety since cyberphysical avatars perform cooperative tasks in close proximity to humans. Neuroevolution technique is essential for "programming" cyberphysical avatars inasmuch as they are to be used by non-experts for a large array of tasks, some unforeseen, in an unstructured environment. QoS-guaranteed real-time communication is essential to provide predictable, bounded-time response in human-avatar interaction. By integrating these technologies, we have built a prototype cyberphysical avatar testbed.**

I. Introduction

The utility of teleoperated robotic devices in mission-critical tasks is undisputed. Despite the impressive progress by the robotics community in recent years, we are still quite a way from being able to trust fully autonomous robots to carry out mission-critical and safety-critical operations by themselves. On the other hand, today's unintelligent teleoperated devices cannot be counted on to perform well in physically difficult and unstructured environments. Short of a gigantic leap in technology that creates intelligent fully autonomous robots capable of functioning in an unstructured environment, we propose to chart a pathway to evolve the capability of teleoperated robotic devices from primitive mechanical remote-control to trustable autonomy and more intelligent teleoperation. Rather than trying to build fully autonomous robots from scratch, we do it gradually through less and less human teleoperation, all the while deploying these robots in actual tasks. This is a new and different approach, and likely to result in practical applications and advances much sooner, and in more robust and better adapted autonomous robots in the end.

This paper chronicles our attempt to explore such a pathway by introducing for the first time the concept of a "cyberphysical avatar". We define a cyberphysical avatar to be a semi-autonomous robotic system that adjusts to an unstructured environment and performs physical tasks subject to critical timing constraints while under human supervision. A cyberphysical avatar is semi-autonomous in that there are actions it must take without human intervention because of the relatively short timing constraints, e.g., the control loop that maintains a fast walking gait. On the other hand, a cyberphysical avatar should not be programmed to deal with only a fixed set of scenarios because we cannot foresee all the contingencies in all operational environments, e.g., a building on fire in a rescue mission. An effective interface between the cyberphysical avatar and its human supervisor is essential for success, and this requires the avatar to be designed for predictable and timely response. As the cyberphysical avatar gains more physical skills, it can be trusted to perform more subtasks on its own.

Our research team consists of roboticists, computer scientists and artificial intelligence researchers who have worked together to define a system architecture for a cyberphysical avatar. There are many technical challenges in realizing the cyberphysical avatar concept. These challenges can be categorized into three topics below:

- **Dynamics and control of humanoid avatars:** The cyberphysical avatar must be able to perform the physical tasks in an unstructured and uncertain environment. In this area, we need to develop a methodology for modeling the dynamic behavior of physical avatars interacting with unstructured environments and controllers that can adapt to the changing physical conditions. We need to develop software foundations that encapsulate the physical skills supporting the full range of teleoperated to autonomous behaviors.
- **Evolutionary learning of skills under environment and performance constraints:** Evolutionary approach is needed to learn the continuous control parameters of the skills, as well as their discrete composition. The cyberphysical avatar must be able to acquire skills so that it can perform time-critical tasks autonomously. The level of autonomy and the criticality of the timing constraints that can be satisfied for practical physical tasks requires advances in learning theory and engineering validation. Moreover, learning strategies need to be applied to learn the tradeoff between teleoperation and autonomy.
- **Supporting reliable, real-time avatar-human communication:** The cyberphysical avatar must be able to operate untethered and maintain timely and reliably communication with the controller that is hierarchically implemented with the human supervisor at the top of the control hierarchy. The combination of real-time and robust communication in a wireless environment where communication paths may be disrupted requires advance in both algorithm design and engineering validation. We need a switching policy between teleoperated and autonomous behaviors that is based on communication quality as the primary metric for making switching decisions.

The rest of this paper will describe in more detail the system architecture of the cyberphysical avatar and the progress we have made in formalizing and partially solving the above technical challenges. We shall describe our prototype implementation using the Dreamer/Meka humanoid robot.

II. Control of wheeled humanoid avatars in unstructured environments

The cyberphysical avatar must be able to maneuver in irregular terrains while performing accurate physical whole-body compliant interactions with the environment and with human operators. To attain these capabilities, skill modeling and control in unstructured environments must be carefully designed. In this section, we first describe the dynamic model of the wheeled base of our Dreamer/Meka humanoid robot under varying contact conditions. We then present our model of the whole-body compliant skill of the robot and the hierarchical control structures that is used to handle task conflicts during the execution of the behavior.

A. Dynamic model of the wheeled base

In unstructured and uncertain environments, wheel-based avatars will often be in a situation of marginal contact, i.e. not all the wheels are in contact. As such, the dynamics of the robot, need to represent the contact state of the robot, its effect on center of mass balance and the conservation of angular and linear momentum due to marginally-stable contact conditions. These characteristics become even more critical when the robot engages into manipulation tasks while maintaining marginal contacts.

We derive the model of the robot under varying contacts by leveraging the generalized contact consistent Jacobian developed in [1] which specifies that for a given contact state C_m the generalized Jacobian of an operational task (e.g. one of the robot's hands) is equal to

$$J^*_{\text{task},C_m} \triangleq J_{\text{task}} \, \overline{UN_{C_m}}, \tag{1}$$

where J_{task} is the Jacobian of the hand Cartesian point with respect to an inertial frame outside of the mobile base, U describes the underactuated (i.e. uncontrollable directions) of the base due to the contact state, N_{C_m} describes the current contact state (i.e. how many wheels are in contact), and the operator $\overline{(.)}$ indicates a dynamically consistent generalized inverse of the argument. Therefore the control of the operational task (e.g. the control of the robot's hand) while taking into account the mobility of the base and the uncertain contact state is equal to

$$\Gamma_{C_m} = J^{*T}_{\text{task},C_m} \, F_{\text{task}} \tag{2}$$

where F_{task} is the force or impedance command to control the hand, J^*_{task,C_m} is the whole-body task Jacobian including the base contact state (i.e. how many wheels are in stable contact), and Γ_{C_m} is the whole-body command of torques sent to the base and upper humanoid torso motors.

B. Skill definition and hierarchical control structure

In whole-body compliant control (WBC), a task is defined via a mapping between the robot's N-dimensional joint configuration and some M-dimensional space which describes an objective that the controller should achieve. The skill is defined as a juxtaposition of multiple operational tasks to help translate between high-level goals (such as provided by planning algorithms) and the operational tasks. In our environment, a skill is a human readable file (e.g. YAML) describing the points or coordinates of the robot that are to be simultaneously controlled to accomplish a behavior, plus their respective control policies, and plus their hierarchical priorities in the execution pipeline. In this work, however, as to be elaborated in Section III, the control policies of the skill will be learned through machine learning approaches.

Having now many operational task processes to simultaneously optimize, as defined in the skill, either as force or impedance processes, we propose to use the following control structure in Eq. 3. The intuition behind this control structure is to instantiate several tasks, each of which tries to drive the robot toward some state. The task contributions are accumulated using null space projections to ensure that lower-priority tasks do not interfere with higher levels. The motion is thus determined by each task in combination with their priorities. This structuring provides two orthogonal ways of changing robot behavior, either by influencing the tasks (e.g. changing their gains or goals) or by rearranging the hierarchy (e.g. inserting tasks or locally inverting their ordering).

$$\Gamma_{C_m} = \sum_k \left(J^{*T}_{k|\text{prec}(k),C_m} F_k \right) + N^{*T}_{t,C_m} \Gamma_{\text{posture}} + J^{*T}_{i|l,C_m} F_{\text{int}}, \tag{3}$$

In Eq. 3, F_k is the task space force or impedance command for the k-th operational task, $J^*_{k|\text{prec}(k),C_m}$ is the prioritized contact consistent Jacobian of the task, Γ_{posture} is the command to optimize the posture behavior, F_{int} is the command to optimize the internal forces between the arms and the mobile base, and $J^*_{i|l,C_m}$ is the Jacobian of the internal forces. This structure is a derivation of our previous work on whole-body compliant control found in [2].

In the control structure, the low-level tasks describing the skill are aggregated using a hierarchy, where more relevant tasks, such as those who ensure the fulfillment of physical constraints appear first, while those dealing with the operational behavior appear with less priority. Fig. 1 gives an example where the skill is composed of three tasks. The first task is maintaining coordinates of center of mass(CoM) to prevent the robot from falling down on irregular terrain. Second task is compliant hand position which enables the robot to respond compliantly to human interaction. The posture task here is utilizing remaining degree of freedom to stabilize self-motion and converge to a human-like posture. Lower-priority tasks operate in the null space of all higher priority tasks. So when the terrain changes the CoM task will temporarily override non-critical tasks in order to prevent falling. The task becomes unfeasible when the current higher priority tasks use all the dynamic redundancy. This event can be easily monitored and used to stop the behavior and communicate the problem to a high level planner.

Because our control structures use effectively the dynamic and contact model of the physical avatar in its environment, they are able to optimize all task processes simultaneously within the contact stance, thus achieving precise tracking of forces and trajectories. Moreover, posture behavior, which is specified as an optimization criterion instead of a trajectory is also optimized within the residual manifolds left over by the priority tasks.

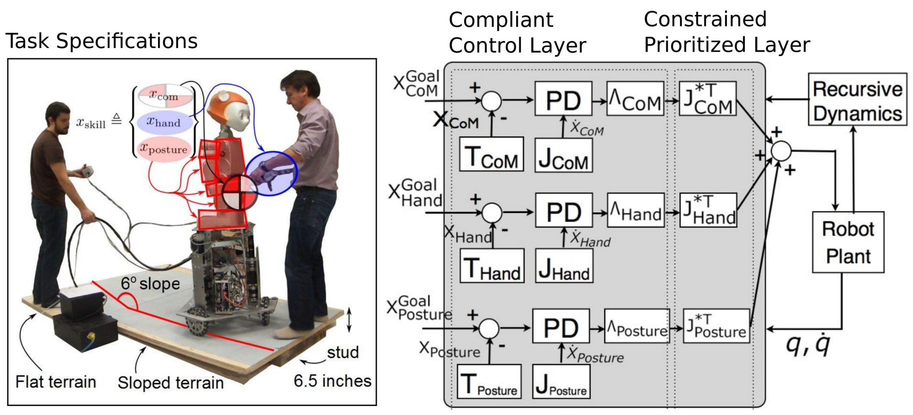

Fig. 1. Whole-body compliant control with prioritized tasks. Left hand side of figure shows the Dreamer crosses a terrain with a slope while responding to human interaction. And right hand side of the figure shows closed loop dynamic controller producing joint torque outputs based on Center of mass, hand position and posture with prioritized Jacobians.

III. Skill acquisition by machine learning

Although much of the operation of the robot can be based on carefully designed control algorithms, there are two issues where machine learning methods can prove crucial: (1) conversion of human operator behaviors to robot behaviors, and (2) optimization of robot behaviors. In both of these cases, it is possible to come up with measures of how good the behaviors are, but the optimal behaviors are not known. Therefore, machine learning methods based on exploration need to be used. In this section, a particularly powerful such a method, neuroevolution, is described first, followed by its application to train the learning skills of the grasper on the Dreamer humanoid robot to pick up objects with any shape.

A. Learning robust nonlinear control through neuroevolution

In the neuroevolution approach, evolutionary optimization method such as a genetic algorithm is used to construct the structure and the connection weights of a neural network so that the network performs as well as possible in a given task [3], [4]. The neural network can be recurrent, implementing a sequence memory, and thereby making it possible to use the approach to discover sequential behaviors such as robot navigation, arm control, and grasping.

Neuroevolution differs from other machine learning methods in two important ways. First, neuroevolution learning is based on exploration and reinforcement: a population of neural networks is evolved through crossover and mutation, directed only by how well each network performs. It is thus possible to discover successful behaviors that human designers would find difficult to construct, and behaviors that are more general. Second, the neural networks employed in neuroevolution can disambiguate the system state based on their sequence memory. Previous sensor values are part of the state representation, making it possible to understand how the world is changing, and how to respond to it optimally. The neuroevolution approach can thus be used as a training mechanism for the Dreamer robot. In particular, it is well suited for learning skills such as picking up an object. In the following, we will first describe the physics of the grasper on the Dreamer/Meka humanoid robot and then present the training details. Training is done following the NEAT [5] approach.

B. Physics of the grasper

We simulate the Meka hand using GraspIt! [6], an open-source grasping simulation environment developed at Columbia University. The Meka hand in GraspIt! is defined by one degree of freedom for each knuckle in each finger, as well as degrees of freedom for the thumb's rotator. The mechanics of this model will be modified in our studies because most degrees of freedom in the Meka hand are not actuated. Each finger consists of three joints, which are all connected by a single rubber tendon. When the finger curls, all three knuckles curl in unison. We will therefore adjust the torques that we feed to the simulator to account for this interdependent joint behavior. A set of torques given to a single finger will conform with one another such that they are all equivalent to torques initiated by a stretching of the rubber tendon, which we see in the real robot.

C. Training the grasper

In order to properly grasp with the Meka hand, we must first design the input and output layers of our target neural network, as well as a fitness function to allow a gradual climb toward an efficient grasp. Because our Meka unit is primarily controlled using the Whole-Body Control framework, the network is only responsible to directly manipulate orientation and positions on the wrist and the finger joints of the hand. Thus we designate an output node for each of these degrees of freedom.

Designing the input layer is less trivial. It is necessary to encode the entire state of the grasp in some way, which includes positions of the hand, the fingers, and grasped object, as well as the object's shape. Proprioception is sufficient for determining joint angles, so for each finger joint we designate a single input node, and we do the same for the wrist rotation and the hand position. To reduce dependency on the shape of the target object, we encode the object's state by simply taking a depth from the robot's Kinect Camera and assigning each depth data a unique input node in the neural network. In this way the net work is able to associate the state of the robot's body and the state of an arbitrary object with an appropriate grasping hand gesture. Evaluation of the grasp is done by combining the grasp quality metrics provided by the GraspIt! and the distance between robot's palm and center of gravity of the object.

D. Transitioning from simulated to physical controller

The training method described in Section III-C is primarily done in simulation. However, transferring controllers evolved in simulation to the physical robot is challenging [7], [8]. The main reason is that it is difficult to simulate physical properties such as friction and sensor and actuator characteristics with high enough fidelity to reproduce the simulated behaviors on real robots. To address this issue, we choose the following methods to improve the results of transfer to the real robot.

First, if the simulator is accurate enough, controllers that transfer well can be created simply by evolving them to be robust. That is, if sensor values and actuator responses frequently vary in simulation, the resulting controllers will be robust against small discrepancies between simulation and reality as well. Such uncertainties can be introduced into the simulation simply as noise, and solutions evolve that do not depend on accurate values and outcomes, thus transferring well.

If there are more systematic flaws in the simulation, behaviors may evolve that exploit them, and therefore transfer poorly. Such behaviors can be discouraged by incorporating transfer into evolution explicitly, by utilizing a multi-objective evolutionary algorithm that optimizes both a task-dependent controller fitness as well as a measure of how well the controller transfers from simulation to reality [9]. In any given generation, this method chooses at most one controller based on behavioral diversity to be evaluated on the real robot, requiring only a small number of hardware evaluations.

Another approach is to perform experiments on the real robot in order to improve the simulator, typically in one of two ways: (1) Experiments are performed on the real robot before running evolution to collect samples of the real world by recording sensor activations [10], [11]. When controllers are evaluated later during evolution, these samples are utilized to set the simulated sensor activations accurately. (2) Experiments are performed on the real robot during evolution to co-evolve the simulator and the controller, making an initially crude simulation more and more accurate [12].

We will adopt the system-level simplex architecture [13] to provide safety guarantees during the transitioning. In this architecture, we use a simple and verified safety controller to ensure the stability and safety of the robot operations. This conservative safety control core is then complemented by a high-performance complex controller, which will be used whenever possible, but switch to the safety controller when system integrity is jeopardized.

IV. Supporting Remote, Reliable, and Real-Time (R^3) Avatar-Human Communication

A key component of cyberphysical avatar technology is reliable and real-time communication between the avatar and remote human supervisor. Because of the mobility requirement of avatars to work in unstructured environments as is often the case in disaster recovery, wireless connection has to be established at the edge of the communication infrastructure. Owing to limited wireless communication range, a multi-hop wireless network in mesh topology is necessary to cover a larger area, and more importantly, can overcome transmission blocking by objects such as metal doors in industrial facilities where avatars operate.

We envision the use of a combination of multi-hop wireless mesh networks and the existing Internet to support real-time communication between the avatars and human supervisor. However, the need for reliable and real-time communication imposes several significant challenges. Specifically, the current Internet architecture cannot guarantee any end-to-end QoS requirement for time-critical control flows and most existing wireless standards and routing protocols are not designed with real-time delay constraint in mind and as a result cannot provide any bound on end-to-end delay. Moreover, the inherent lossy wireless medium, the constantly fluctuating traffic volumes and channel conditions, together with complicated interference relationship have made it challenging to achieve high end-to-end reliability, which is essential for remote avatar-human communication.

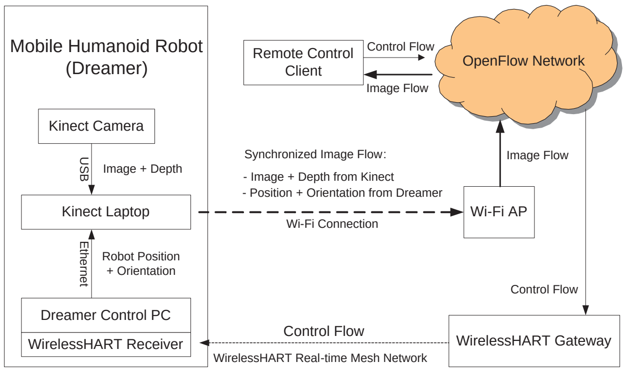

Fig. 2. The R^3 communication infrastructure for cyberphysical avatars

A cyberphysical avatar typically contains two types of data flows. There is one or multiple data flows destinated to remote human supervisor containing physical information of the environment in which the avatar operates. These data flows include image flows captured by the cameras installed either on the avatar or in the operating environments, and real-time position/direction information of the avatar. There is also a control flow originated from the human supervisor to control the robot and finish designated tasks. In our R^3 communication infrastructure, we use Wi-Fi connection for transmitting data flows because they usually require larger bandwidth but have soft real-time requirements on packet delivery. On the other hand, the time-critical control flow is transmitted on WirelessHART [14] real-time mesh network which is set up at the edge of the communication infrastructure to guarantee the end-to-end delay in the avatar-human communication. OpenFlow [15] network are deployed to enhance the existing Internet architecture to provide better QoS support. Figure 2 summarizes our communication infrastructure, and it consists the following key components.

A. Wi-Fi connection for supporting data flows

We set up a Wi-Fi connection at the edge of the communication infrastructure to help forward data flow to the remote human supervisor. In our current setting, the data flow is an image stream captured by the Kinect sensor installed on the avatar. Both the color images and depth images from the Kinect sensor will be synchronized with the position and orientation information received from the avatar control PC and sent to the remote supervisor. the remote control application receives the images and renders them on the user interface. It allows the supervisor to monitor the physical environment the avatar is operating in and supervise it to execute designated tasks.

B. OpenFlow network for providing QoS guarantees

To meet the strict end-to-end delay constraint and jitter requirement on the avatar-human communication, guaranteeing the QoS requirement of the control and image flow on the Internet should be achieved. However, this is not a trivial problem because the current Internet architecture has no facility to guarantee minimum bandwidth and end-to-end latency for network flows. To address this problem, we are deploying OpenFlow [15] switches to connect remote human supervisor and the avatar operating environments. We use Beacon [16] OpenFlow controller to set up a particular queue for the control flow in the OpenFlow switch. For the image flow of cyberphysical avatar system, we setup minimum rate queue that the rate for image flow is guaranteed to be large enough.

C. WirelessHART mesh for supporting real-time control flow

The control flow from the remote human supervisor to the robot control PC is time-critical and has hard deadline on its delivery. Due to the pervasive Wi-Fi signals and the backoff mechanism used in 802.11, the jitter in Wi-Fi transmission is large and unpredictable. This is a fatal disadvantage of Wi-Fi to be adopted for providing reliable and real-time communication for the control data flow in cyberphysical avatars.

For this reason, in the R^3 communication infrastructure designed for cyberphysical avatars, we use WirelessHART [17] real-time mesh network to replace Wi-Fi for transmitting the control data flow to the avatar control PC. To improve the service scalability and make the application development easier, we further enhance the communication stack and the Gateway with a 6LoWPAN adaptation layer, a UDP transportation layer and a CoAP application layer. This enhancement makes the WirelessHART device IP-enabled and the Gateway only needs to take the role of the router and forward the IP packets to the avatar control PC. Only the remote control application and the application running on avatar control PC need to understand the specific application protocol. The Gateway can remain unchanged when new services are established between the supervisor and the robot.

V. Designing and building a cyberphysical avatar

We are building a cyberphysical avatar to verify the effectiveness of the proposed architecture. The remote control application is installed in UT ACES building and the Dreamer robot is located in UT Human Centered Robotics Lab. OpenFlow switches are being deployed in UT campus network to provide QoS guarantee especially for the control data flow between the remote control application and the avatar control PC. In this section, we will present the details of the system setup and give a demo of remotely supervising the Dreamer robot to execute specific tasks [18].

A. System setup in Human Centered Robotics Lab

Figure 3 describes the system setup in the Human Centered Robotics Lab in UT mechanical engineering department. There are three key components in the system setup:

The Dreamer/Meka Hardware: The main hardware tool that we use for this study is the Dreamer/Meka mobile dexterous humanoid robot. This robot includes the T2 Meka torso, the A2 Series Elastic Meka arm, the H2 tendon driven Meka hand, the Dreamer/Meka head co-developed by Meka and UT Austin, and the torque-controlled holonomic UT Austin's Tricky base. The actuators for the base and the upper body, except for the head, contain torque/force sensors that enable Elmo amplifiers to implement current or torque feedback. An Ethercat serial bus

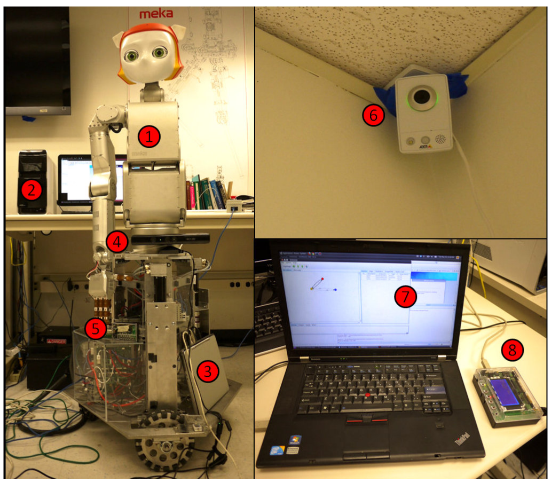

Fig. 3. An overview of the system setup in UT Human Centered Robotics Lab. (1) The Dreamer robot. (2) Robot control PC. (3) Kinect Laptop. (4) Kinect sensor. (5) WirelessHART receiver. (6) IP camera. (7) WirelessHART Gateway. (8) WirelessHART access point.

communicates with sensors and motor amplifiers from a single computer system. A PC running Ubuntu Linux with the RTAI Realtime Kernel runs the models and control infrastructure described in this project. The Tricky holonomic base contains torque sensors as well as the inertial measurement unit (IMU) 3DM-GX3-25 from MicroStrain. It achieves holonomic motion and force capabilities by utilizing Omni wheels located in a equilateral triangular fashion.

Kinect/IP Cameras: We have two cameras installed in the Human Centered Robotics Lab. An IP camera is installed at the right upper corner to give an overview of the avatar environment and a Kinect camera is installed on the Dreamer robot to capture the image and depth information of the target. Due to the limitation of the power and the computation capability, the Kinect camera is installed on a separate Laptop which is put on the Dreamer base and connect to the avatar controller through Ethernet. The Kinect Laptop synchronizes image streams (including the image and depth information) captured from the Kinect camera and the Dreamer position and orientation information together and sends to the remote supervisor through a Wi-Fi Access Point connected to the campus network.

WirelessHART real-time communication subsystem: A WirelessHART Gateway/Network Manager is set up and connected to the UT campus network. The Gateway contains an access point through which WirelessHART devices can join and talk to the Gateway/Network Manager. For simplicity, we have one WirelessHART device connected to the avatar control PC through serial port and form a one-hop communication. More devices can be deployed to form a mesh network to cover larger area if necessary. The WirelessHART device exchanges the control commands and robot status with the avatar control PC through shared memory. The robot status information is transmitted back to the remote supervisor on WirelessHART real-time wireless network in the reversed direction.

B. Remote Control Application

Figure 4 shows a screen capture of the remote control application we developed for supervising the Dreamer robot. The details of the interface messages between the control

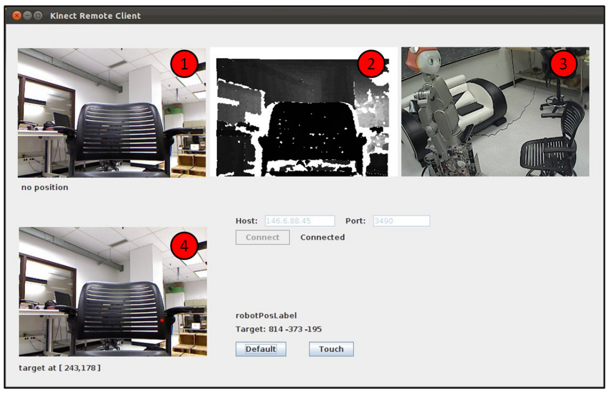

Fig. 4. A screen capture of the remote control application for supervising the Dreamer robot. (1)(2) Color and depth image from Kinect sensor. (3) Image from IP camera. (4) Image snapshot when user presses the color image.

application and the Kinect Laptop/Dreamer robot can be found in the technical report [19].

The specific skill we are training the Dreamer robot is to move itself close to a desk and pick up a designated target under supervision. As shown in Fig. 4, in the remote control UI, the color and depth images from Kinect camera and the images from the IP camera are displayed in the three image panels at the top of the UI. The human supervisor can choose a target in the color image from Kinect panel by clicking on it. After clicking on the target, a copy of the color image at this moment is copied to the image of target object panel. A red dot is added on the image showing the position the user clicked. The coordinate of this position is also displayed in the mouse clicked position label. Using the position of the click on the image and the depth data at that point, we calculate the physical coordinate of the target with respect to the Kinect. Once get these information, user can issue commands to the Dreamer robot to execute specific tasks. In our current testbed, two commands have been implemented already: Default and Touch. The Default command gets the robot back to the default gesture, while the Touch command asks the robot to touch the target we clicked on. A sequence of video snapshots is presented in the technical report [19] to demonstrate an user issued Touch commands to the robot. It is possible that the Kinect sensor has small measurement error due to its hardware limitation. In that case, we relied on visual feedback, and used incremental move commands to direct the robot to touch the target object. The grasp skill to lift up the target is under training and the Grasp command will be added to the remote control application as soon as the training is finished.

VI. Conclusion and Future works

This paper introduces the concept of a cyberphysical avatar which is defined to be a semi-autonomous robotic system that adjusts to an unstructured environment and performs physical tasks subject to critical timing constraints while under human supervision. A cyberphysical avatar is the bridge technology that will help transition dumb teleoperated robotic devices to autonomous robots capable of functioning in unstructured environments. What makes the cyberphysical avatar possible today is the convergence of three recent technologies: body-compliant control in robotics, neuroevolution in machine learning and QoS guarantees in real-time communication. By integrating these technologies, we have built a prototype cyberphysical avatar testbed. Building this physical testbed is an essential first step for exploring the cyberphysical avatar concept because the physical and computational complexities involved necessarily require less than exact modeling and the use of mathematical approximations to simulate physical processes; the viability of the cyberphysical avatar must be validated by a physical testbed. As of this time, the basic body-compliant controller and the communication subsystems have been integrated; work is ongoing to improve the interface between the robot and the controller and to train the robot by the NEAT [5] algorithm.

Acknowledgment

We thank Hongkun Yang, Quan Leng and Sangki Yun for their help in developing the remote control application, and we are grateful to Meka robotics for their technical support.

References

[1] L. Sentis, *Motion Planning for Humanoid robots*, chapter Compliant Control of Whole-Body Multi-Contact Behaviors in Humanoid Robots, pp. 29–63, Springer Berlin Heidelberg, 2010.
[2] L. Sentis, J. Park, and O. Khatib, "Compliant control of multi-contact and center of mass behaviors in humanoid robots," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 483–501, June 2010.
[3] Risto Miikkulainen, "Neuroevolution," in *Encyclopedia of Machine Learning*. 2010.
[4] Dario Floreano, Peter Dürr, and Claudio Mattiussi, "Neuroevolution: From architectures to learning," *Evolutionary Intelligence*, vol. 1, pp. 47–62, 2008.
[5] Kenneth O. Stanley and Risto Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.
[6] "GraspIt!," `http://sourceforge.net/projects/graspit/files/releases/`.
[7] Nick Jakobi, *Minimal Simulations for Evolutionary Robotics*, Ph.D. thesis, School of Cognitive and Computing Sciences, University of Sussex, 1998.
[8] Hod Lipson, Josh Bongard, Victor Zykov, and Evan Malone, "Evolutionary robotics for legged machines: From simulation to physical reality," in *Proceedings of the 9th International Conference on Intelligent Autonomous Systems.*, 2006, pp. 11–18.
[9] Sylvain Koos, Jean-Baptiste Mouret, and Stéphane Doncieux, "Crossing the reality gap in evolutionary robotics by promoting transferable controllers," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*. 2010, pp. 119–126, ACM.
[10] Orazio Miglino, Henrik Hautop Lund, and Stefano Nolfi, "Evolving mobile robots in simulated and real environments," *Artificial Life*, vol. 2, pp. 417–434, 1995.
[11] Stefano Nolfi, Dario Floreano, Orazio Miglino, and Francesco Mondada, "How to evolve autonomous robots: Different approaches in evolutionary robotics," in *Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems (Artificial Life IV)*.
[12] Juan Cristóbal Zagal and Javier Ruiz-Del-Solar, "Combining simulation and reality in evolutionary robotics," *Journal of Intelligent and Robotic Systems*, vol. 50, pp. 19–39, 2007.
[13] S. Bak, D.K. Chivukula, O. Adekunle, Mu Sun, M. Caccamo, and Lui Sha, "The system-level simplex architecture for improved real-time embedded system safety," in *Real-Time and Embedded Technology and Applications Symposium*, april 2009, pp. 99 –107.
[14] "WirelessHART," `http://www.hartcomm.org/protocol/wihart/wireless_technology.html`.
[15] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner, "Openflow: enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, 2008.
[16] "Beacon," `https://openflow.stanford.edu/display/Beacon/`.
[17] Jianping Song, Song Han, Al Mok, Deji Chen, Mike Lucas, Mark Nixon, and Wally Pratt, "WirelessHART: Applying wireless technology in real-time industrial process control," in *Proc. of IEEE RTAS*, 2008.
[18] "Avatar Demo," `http://www.cs.utexas.edu/~shan/avatar.html`.
[19] Song Han, Aloysius K. Mok, Jianyong Meng, Yi-Hung Wei, Xiuming Zhu, Luis Sentis, Kwan Suk Kim, Risto Miikkulainen, and Jacob Menashe, "Architecture of a cyberphysical avatar," *UTCS Technical Report #TR-12-12. http://apps.cs.utexas.edu/tech_reports/reports/tr/TR-2082.pdf*.