Real Time Human Motion Imitation of Anthropomorphic Dual Arm Robot Based on Cartesian Impedance Control

Ren C. Luo, Bo-Han Shih, Tsung-Wei Lin
International Center of Excellence on Intelligent Robotics and Automation Research
National Taiwan University
Taipei, Taiwan
renluo@cc.ee.ntu.edu.tw

***Abstract*—This paper presented a real-time human motion imitation approach to control an anthropomorphic dual arm robot by human demonstration. We use the processed positions of human skeleton joints from Kinect sensor as commands directly to control the robot arms by using Cartesian impedance control to follow the human motion without solving inverse kinematics problem. In order to avoid a jerky robot arm motion, we apply an on-line trajectory generator algorithm to obtain a smooth movement trajectory by imposing the limit of velocity and acceleration. Moreover, the self-collision problem has also been considered. When the distance between two parts of body is close enough, a repulsive force will automatically generate to prevent collision. Taking the robot capability and safe issue into account, the output force is restricted to ensure that the action of robot is stable. We demonstrate the feasibility of the approach by implementing the human motion imitation system on a humanoid dual arm robot developed in our lab. The experimental results show that the system is in good practice and flexible enough to imitate various human motions.**

***Keywords*—human motion imitation; self-collision avoidance; Cartesian impedance control**

I. INTRODUCTION

In recent years, the development of human robot interaction in service robots has attracted the attention of many researchers. In order to let humanoid robots have similar behavior with human, motion planning is crucial. However, planning human-like motions for robots is very complicated because it needs to handle multiple degrees of freedom (DOFs) simultaneously. Learning by demonstration is an intuitive and efficient way to let a humanoid robot learn a variety of human-like motions. The idea is to generate human-like motions by extracting information directly from human motion via a motion capture system, and it simplifies the process of programming and learning complex motions.

Many applications of imitation system have been tested by using a variety of type and size of humanoid robots. A real-time human mimicking system based on NAO with the ability of balance control and self-collision avoidance is introduced in [1]. A simulation of imitation system experimented by a small-size DARwIn-OP humanoid robot is presented in [2]. An integration work of physical human robot interaction into imitation learning is implemented by using the IRT humanoid robot and DLR's humanoid upper-body robot Justin [3]. A method of smooth transition between tasks on a kinematic control level for self-collision avoidance application is demonstrated on two KUKA LWR robots [4].

A lot of researches have been proposed about human motion capture through different sensor devices, e.g. cameras, depth sensors, inertial sensors or marker based vision system [5]-[7]. In our work, we use the Kinect sensor developed by the Microsoft as our motion capture device. Kinect sensor composed of a RGB camera and a depth sensor is widely used in motion capture system recently because of its relative low price. The basic technique of the depth sensor is to project a structured infrared light continuously and calculate depth from the reflection of the light at different positions. By processing the depth image, user's skeleton joints can be captured and provided in real time.

However, human mimicking system is not merely a simple "copy" operation because of different kinematic structures between human and robot. The link length and joint configuration are not the same as well as some constraints imposed by robot mechanism. Several researches have been developed for converting human poses into suitable robot joint commands for a humanoid robot [8]-[10].

In this paper, our main objective is to show that the robot can imitate the human arm motion by human demonstration in real time (see Fig. 1). That is, human arm movement will be directly reflected in the action of robot arms. If the human arms move quickly or suddenly change the speed and direction, it will cause the robot arm jerky motions and such behaviors

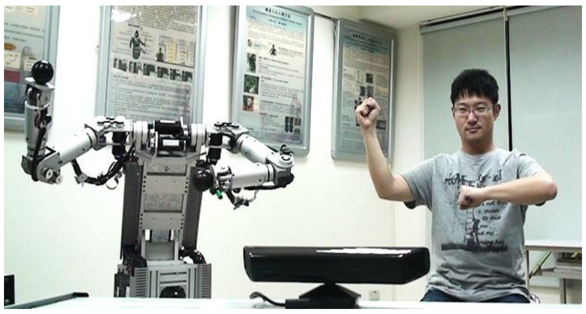
Fig. 1. The robot is controlled by human demonstration in real time using Kinect sensor.

might not only damage the robot itself but also give people an unsafe feeling. To avoid such situations, we use an on-line trajectory generator [11], [12] to limit the change of robot arm speed with velocity and acceleration constraints, so that the smooth robot arm movement trajectories are generated.

In the aspect of robot control, we use Cartesian impedance control to command the robot arms tracking. The main idea is to select a set of control points on the robot and then these control points are virtually connected to corresponding command points via virtual springs and damping elements in XYZ directions respectively. When the command points move in space, the robot arm will be guided by the forces generated from these springs. In this way, robot can follow the human motion without the need of explicitly computing inverse kinematics. Taking the self-collision problem into consideration, a self-collision avoidance scheme has been integrated into the system. In addition, to ensure the robot arm move stable and safe, the increasing rate and an imposing upper bound of the output force are required. In this research, we use the method of vector projection to transform Cartesian force into joint torque [13], and then through the torque control loop to drive the robot arm.

The paper is organized as follows. Section II provides an overview of the human motion imitation system including the idea of human motion following and main procedure of the system. On-line trajectory generator for robot imposed with the limit of velocity and acceleration is presented in section III. The architecture of robot arm control which consists of Cartesian impedance and damping control, self-collision avoidance and constrained force limit is described in section IV. Hardware system of the humanoid 6-DOFs dual arm robot and experimental results are discussed in section V. Finally, the paper is concluded with a brief summary in section VI.

II. OVERVIEW OF HUMAN MOTION IMITATION SYSTEM

We employ the Kinect sensor as our motion capture device to track the position of human skeleton joints in real time (see Fig. 2). The frame rate of Kinect sensor is about 30 frames per second. These skeleton joints can be acquired in the form of Cartesian space under Kinect coordinate frame. However, the mapping from human arm posture into robot arm posture is not quite straightforward due to different kinematic structures of link length and joint configuration. In [3] and [4], they find their robot joint angles by applying inverse kinematics method. Since Kinect sensor only provides the position of each joint, we merely know the direction of each link but not the rotation

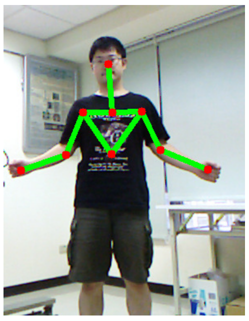

Fig. 2. Human skeleton joints are shown as red dots.

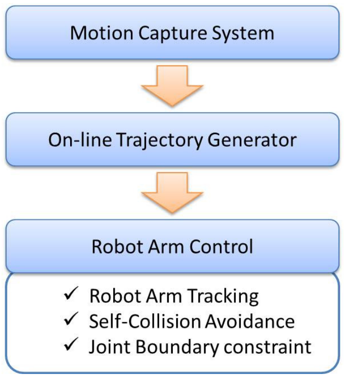
Motion Capture System
On-line Trajectory Generator
Robot Arm Control
✓ Robot Arm Tracking
✓ Self-Collision Avoidance
✓ Joint Boundary constraint

Fig. 3. The main processes for human motion imitation system.

along the link. As a result, it causes that some human postures may correspond to multiple configurations of robot joint. Taking the continuous robot motion into account, an optimal selection for suitable joint angles is required.

In this paper, we demonstrate a method based on control architecture without calculating complex inverse kinematics to make the robot arm follow the posture of human arm. The control of robot arm tracking is implemented under Cartesian space, so we can manipulate the robot arm movement by using Cartesian position command. In order to control the robot arm posture similar to human, we choose the position of elbow relative to shoulder and the position of elbow relative to elbow as our commands. Therefore, we separate the arm tracking into two parts: one is tracking the elbow position relative to shoulder and the other one is tracking the hand position relative to elbow.

After transforming the coordinate from the Kinect frame to the robot frame, owing to the different link lengths between human and robot, the corresponding positions of elbow and hand on the robot need to be calculated according to the link length of robot. In fact, if we deem that the length of the human upper arm is a constant, the scope of activities of elbow relative to shoulder is a spherical surface. In the same reason, the scope of activities of hand relative to elbow is also a spherical surface. In spherical coordinates system, the position of a point is specified by a radius and two rotation angles. In our case, since the link length is fixed, the point on the spherical surface can be decided by two rotation angles. Hence, we command four joints to manipulate one robot arm, first two joints for tracking elbow position and the other two joints for tracking hand position.

The main processes for our proposed human motion imitation system are shown in Fig. 3. First, the motion capture system provides the tracking targets for the robot to follow. Second, by means of the trajectory generator, we can avoid the change of tracking targets too fast and get robot arm moving trajectory much smoother. Finally, we use Cartesian impedance control to track the command trajectories and apply the concept of virtual spring to generate repulsive force to prevent self-collision and also restrict the joint boundary to prevent the damage of the robot. By proceeding the aforementioned processes, a real-time human mimicking system by a humanoid dual arm robot can be achieved. The details of each subject process are described in the following sections.

III. On-Line Trajectory Generator

The frame rate of Kinect sensor is high enough to display the human skeleton in real time, after processing, we could use the data directly as input command to manipulate the robot arm. The action of robot arm can move according to the motion of human arm faithfully if the capability of robot arm is good enough to handle the high speed movement. However, in the general case, some motions might be critical, such as quickly stop or suddenly change the speed and direction, and it will cause the jerky motions of robot arm. Such behaviors not only may damage the robot itself but also give people an unsafe feeling. To overcome such situations, we use an on-line trajectory generator to limit the change of the robot arm speed with velocity and acceleration constraints

At the beginning, we sample the captured data by Kinect sensor and perform the linear interpolation to get the intermediate values between two sampling frames. Because the control timer in our robot arm system is 1 millisecond, we have to generate the movement trajectories per millisecond for the robot arm tracking control. By means of one sampling time delay, we can use equation (1) to calculate via points that are target points for the robot arm to follow.

$$X_{target}(t) = X_{kinect}(t_{i-1}) + \frac{X_{kinect}(t_i) - X_{kinect}(t_{i-1})}{\Delta t_{kinect}}(t - t_{i-1}) \quad (1)$$

The sampling time we access data is Δt_{kinect}. $X_{kinect}(t_i)$ is the newest updated sampling Kinect data and $X_{kinect}(t_{i-1})$ is the last sampling Kinect data. $X_{target}(t)$ is generated per millisecond with linear interpolation. If this target point can be reached by robot arm under constrained velocity and acceleration, it will be the new input command, but if not, we will find the new command conforming to the constraints.

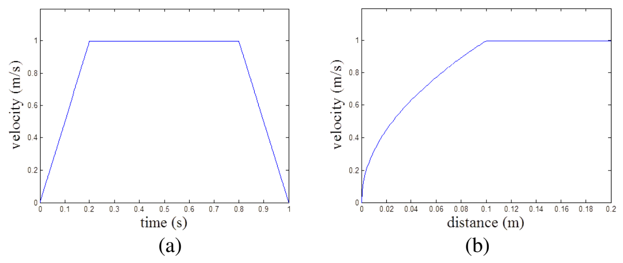

Fig. 4. (a) The relationship between velocity and time. (b) The relationship between velocity and distance from command to target point.

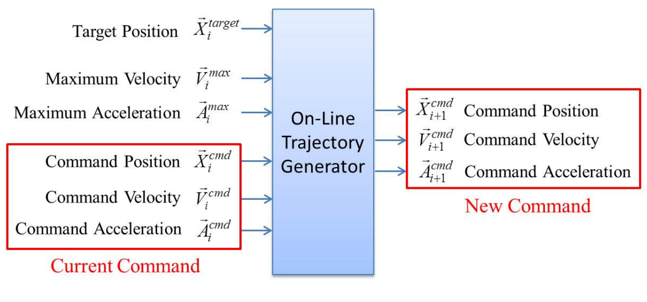
Target Position $\vec{X}_i^{target}$
Maximum Velocity $\vec{V}_i^{max}$
Maximum Acceleration $\vec{A}_i^{max}$
Command Position $\vec{X}_i^{cmd}$
Command Velocity $\vec{V}_i^{cmd}$
Command Acceleration $\vec{A}_i^{cmd}$
Current Command
On-Line Trajectory Generator
$\vec{X}_{i+1}^{cmd}$ Command Position
$\vec{V}_{i+1}^{cmd}$ Command Velocity
$\vec{A}_{i+1}^{cmd}$ Command Acceleration
New Command

Fig. 5. The interface of the on-line trajectory generator.

The trajectory design for the robot arm to follow is based on the principle of constant acceleration motion. A maximum velocity and acceleration are set to avoid the robot arm moving too fast or changing too rapidly. Using equations (2) and (3), we can estimate the command velocity V_{i+1}^{cmd}.

$$s = X_i^{target} - X_i^{cmd} \quad (2)$$

$$V_{i+1}^{cmd} = \frac{s}{\Delta t} \quad (3)$$

s represents the distance between the target point and the command point. Δt is the time interval between two input commands. The command velocity V_{i+1}^{cmd} is restricted by three conditions below.

$$\left| V_{i+1}^{cmd} \right| \le V^{\max} \quad (4)$$

$$\left| V_{i+1}^{cmd} - V_i^{cmd} \right| \le A^{\max} \cdot \Delta t \quad (5)$$

$$\left| V_{i+1}^{cmd} \right| \le \sqrt{2 A^{\max} s} \quad (6)$$

First of all, velocity is limited by maximum value $V^{\max}$, and acceleration is also limited by maximum value $A^{\max}$. When the command point is close enough to the target point, i.e. s becomes smaller, as the result of (6), the velocity will slow down to zero. Through the three constraints, the relationship between command velocity and time will be a trapezoid shape as shown in Fig. 4 (a) when control the robot arm to move from one point to another. At first, the command velocity will speed up with the maximum acceleration until attain the maximum velocity. Before reaching the target point, the command velocity will slow down because the constraint (6). Finally, the command velocity will get to zero when command point reaches the target point (see Fig. 4 (b)). It is possible that the velocity does not reach maximum value, if the initial distance between command point and target point is not far enough. After confirming the new command velocity V_{i+1}^{cmd}, the new command position X_{i+1}^{cmd} also can be known by (7).

$$X_{i+1}^{cmd} = X_i^{cmd} + \tfrac{1}{2}(V_i^{cmd} + V_{i+1}^{cmd}) \cdot \Delta t \quad (7)$$

Based on the current command, limiting conditions of velocity and acceleration, and the target position, a new command can be obtained. In this way, the smoother robot arm movement trajectories are generated. The interface of the on-line trajectory generator is shown in Fig. 5

IV. Control Architecture of Robot Arm

The overall control scheme for the robot arm tracking is shown in Fig. 6. X_i^{cmd} is the Cartesian position command generated by on-line trajectory generator. We use Cartesian impedance and damping control to command the robot arms tracking. The force limit is to restrict the increase rate of the output force for safety. Followed with a method of vector projection to transform Cartesian force into joint torque is used. Finally, a torque control loop has been implemented to drive the robot arm motion.

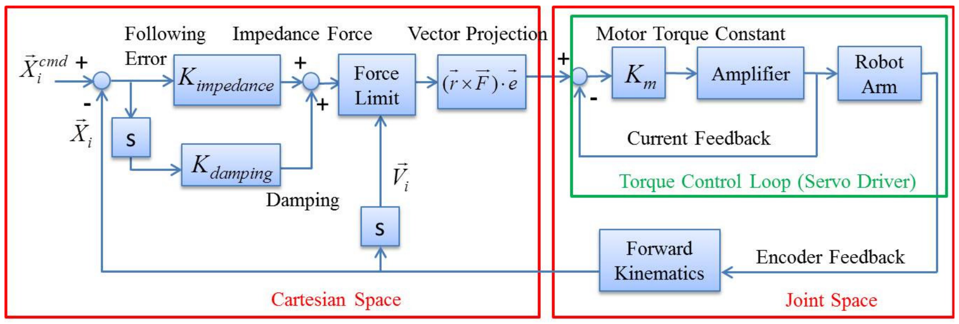

Fig. 6. Overall control scheme for the robot arm tracking.

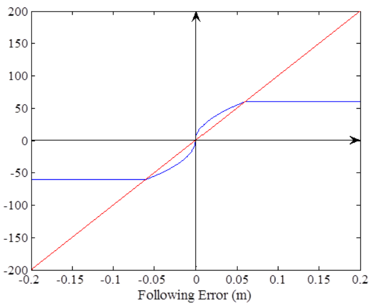

Fig. 7. The limit for output force.

A. *Cartesian Impedance and Damping Control*

We can image that there are virtual damped springs connecting between the command points and corresponding points. Accordingly, when the command points move in space, the robot arm will be guided by the forces generated from these springs. The points we give the robot arm to follow are the positions of elbow and hand on the robot. The force composed of impedance force and damping force on each point is separated into three directions along the Cartesian space XYZ-axis respectively (see Fig. 8). The impedance force is generated by the impedance gain and the following error. And the damping force is expressed as damping gain multiplied by relative velocity. The damping involved in the system is to reduce the overshoot phenomenon and increase the stability when the robot arm moves.

B. *Self-Collision Avoidance*

The occurrence of self-collision can be divided into two cases in our humanoid upper-body robot. One is two robot arms collision and the other is the robot arm bumps into the torso. Virtual spring concept is applied to let robot automatically generate a repulsive force to prevent collision and do not change the trajectory of the arm. We model the links of the robot by the line segments and use a rectangular plane to represent the torso of the robot. Through the forward kinematics, we know the position of the line segments in the Cartesian space and know the relative location between line segments and the plane. By continually examining the minimum distance between two arms' line segments and all the distances from line segments to the plane, we can find out whether there will be a collision occurs. A safe distance threshold is set to avoid the collision.

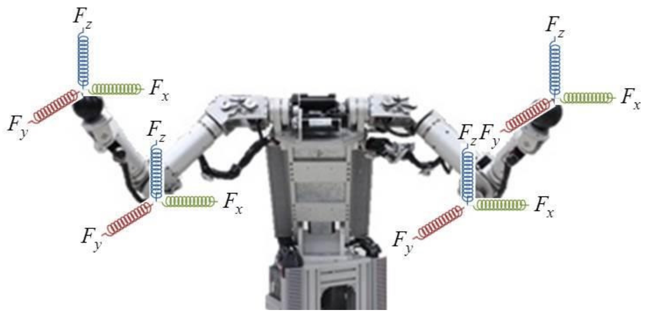

Fig. 8. Virtual damped springs are connected between the command points and corresponding points on the robot in XYZ direction respectively.

As long as a distance is less than the setting threshold, a virtual spring will appear in that position and the virtual spring will generate a repulsive force to repel two parts. The magnitude of the repulsive force is according to the difference between the distance and the threshold, so the smaller the distance, a larger repulsive force will be generated. If there are more than one distance less than the threshold, several virtual springs will appear simultaneously and the direction of the avoidance will be determined by the composition of forces.

The advantage of this collision avoidance method is not to consider the complex path planning of the robot arm but issuing repulsive force to decide the way to go. However, there is also a drawback because we do not know the exactly avoidance behavior. Also because we do not change the robot arm trajectory but forcibly change the position of the robot arm, it could cause a suddenly jump motion when the repulsive force disappears. Therefore, we design a restriction to limit the increasing of the output force and set an upper bound to ensure the robot arm move stable and safe.

C. *Force Limit*

A feature of impedance control is that the system will generate lower stiffness response as robot arm closes to the target position. But if the impedance gain is constant, the force will grow with the following error increases as shown in the red line of Fig. 7. Furthermore, the following error could suddenly become large due to external interference, and if the output force also increases with the following error abruptly, this will lead to robot arm a jump motion. In the application of human robot interaction, such robot arm action is very dangerous, and gives people a feeling of fear. Therefore, a restriction below is designed to adjust the output force.

$$|F_i| \le F_{saturation} \tag{8}$$

$$|F_i| \le \sqrt{c(X_i^{cmd} - X_i)} \tag{9}$$

$$|F_i| - |F_{i-1}| \le \Delta F \tag{10}$$

By (8) and (9), the limit for output force is as shown in the blue line of Fig. 7. A saturation value, $F_{saturation}$ is set to avoid the output force become too large. The parabolic curve part is the result of (9). $X_i^{cmd} - X_i$ means the following error. c is an adjustable coefficient to decide the rising rate. The advantages of this design are that a higher gain to help precise positioning when robot arm closes enough to the target point, and output force will be restricted by an upper bound even if the following error suddenly jumps to a large value. The increasing rate of the output force is restricted by (10) and its intention is to avoid the robot arm in generating too large acceleration, but deceleration is not restricted.

V. Experiments

A. Hardware System

We build a humanoid 6-DOF dual arm robot (see Fig. 9) to implement the experiment and the mechanical specifications of robot arm are shown in TABLE I. In our experiment, we only use the first four joints and the remaining joints are not commanded. The PC-based control algorithm is written in C language. We apply torque control mode in the control algorithm. The torque commands are sent to the servo drivers through the D/A converter and encoders feedback by servo motors. The sampling time in our control loop system is 1 millisecond. The architecture of the robot arm system is shown in Fig. 10.

B. Experimental Results

We test the human motion imitation system with conductor motion and a sequence of photos is shown in Fig. 13. It demonstrates that the humanoid robot arm can be controlled by human demonstration in real time by using Kinect sensor. Fig. 11 records motion trajectories of left elbow position relative to shoulder under Cartesian space respectively in 10 seconds. The line of Kinect means the trajectory of human motion captured by Kinect, the line of OTG means the command trajectory for robot arm provided by on-line trajectory generator and the line of actual means the trajectory of actual robot arm motion. From the results, we can see that the robot arm can follow up human motion in good performance.

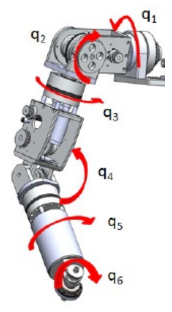

Fig. 9. Kinematic structure of the robot arm.

TABLE I. Specifications of the robot arm.

Degrees of Freedom (DOFs)	Shoulder: 3 DOF (Roll, Pitch, Yaw rotation) Elbow: 1 DOF (Curving elbow) Wrist: 2 DOF Total: 6 DOF
Dimension	Shoulder length: 175 (mm) Upper arm length: 300 (mm) forearm length: 300 (mm) fist length: 7 (mm)
Weight (include mechanism)	Shoulder: 2.3 (kg) Upper arm: 3.3 (kg) forearm: 2.2 (kg) fist: 0.5 (kg) Total: 8.3 (kg)

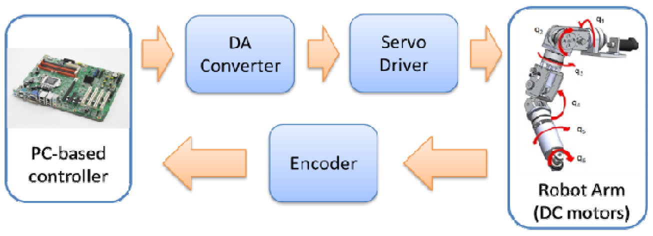

Fig. 10. The architecture of the robot arm system.

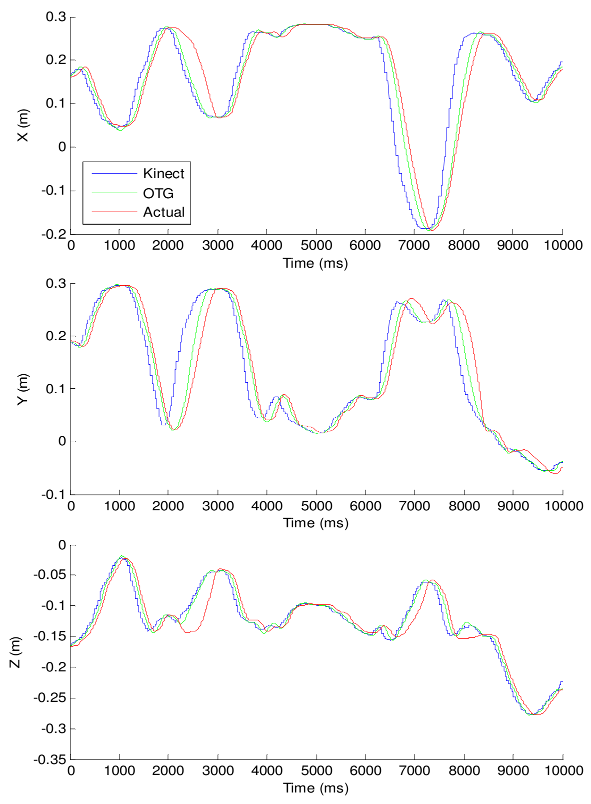

Fig. 11. The trajectory of left elbow in XYZ directions.

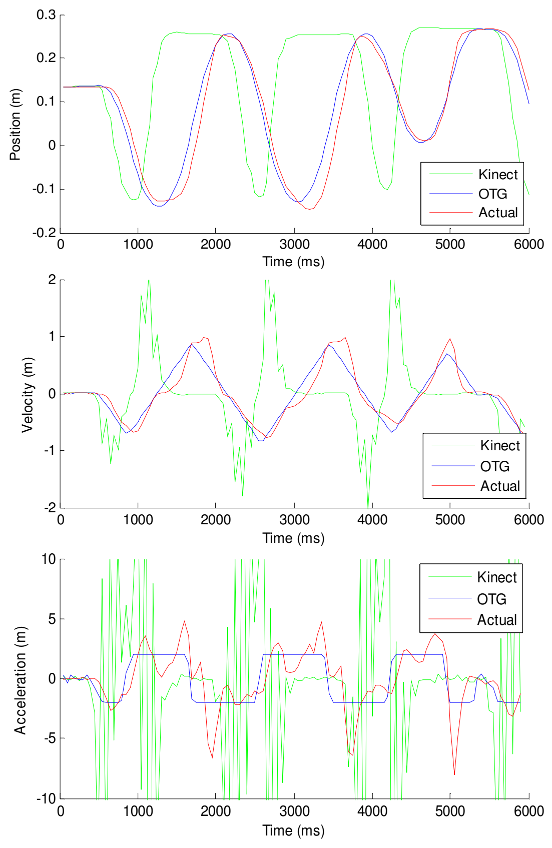

Fig. 12. The trajectory of left elbow with respect to position , velocity and acceleration.

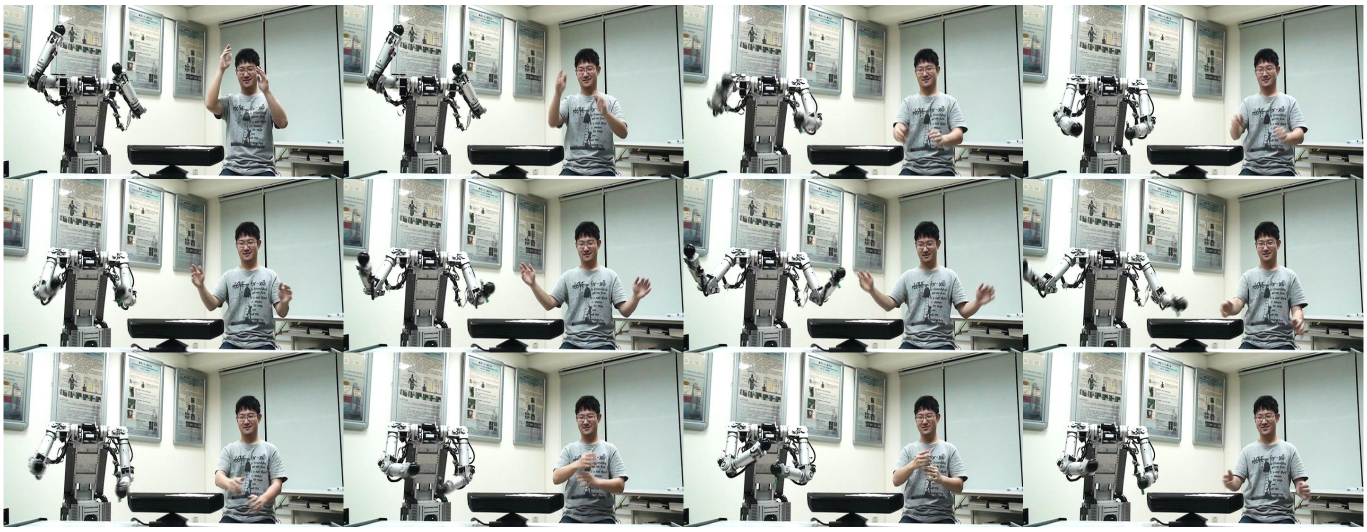

Fig. 13. A sequence of photos shows the robot imitates human motion in real time. (Take three shots per second.)

We try to move arm quickly to observe the reaction of robot, and verify the effect of OTG. Fig. 12 (a) is the trajectory of left elbow in X-axis and the corresponding velocity and acceleration trajectories are shown in (b) and (c) respectively. We can see that the acceleration trajectory of Kinect is vibration dramatically and the velocity trajectory is also vibration and rough significantly. In order to get a smother and safe trajectories, we limit the maximum velocity of the robot arm is 1 (m/s) and the maximum acceleration is 2 (m/s^2). As long as the change rate of Kinect trajectory exceeds the velocity and acceleration constraints, the effect of OTG can be seen. In Fig. 11 (c), the acceleration trajectory of OTG is limited by a boundary, so the change of velocity is at a constant rate and it causes the shape of velocity trajectory is like a triangle wave (see Fig. 11 (b)). After through the OTG, we can get continuous and smooth trajectories. However, it is inevitable that the OTG position trajectory may different with Kinect trajectory. Since the velocity and acceleration is restricted, the response of OTG trajectory is slower than Kinect. In addition, the actual trajectory is in accordance with the OTG trajectory in general cases and it indicates that the human motion imitation system is achieved

VI. CONCLUSIONS

In this paper, we design the frame work of dual arm human motion imitation system based on an online trajectory generator with impedance control, force limit, and self-collision avoidance, and implement the system with successful demonstration. Since the system and all of the functions are well integrated, the robot arms can imitate the upper body motion of a human user immediately, smoothly, and safely.

In the near future, we are going to add online obstacle avoidance functions into the system, making the dual arm robot not only able to avoid hitting other people or objects but also keep a reasonable and smooth motion even though the trajectory is changed.

REFERENCES

[1] F. Wang, C. Tang, Y. Ou, and Y. Xu, "A real-time human imitation system," World Congress on Intelligent Control and Automation (WCICA), 2012, pp. 3692-3697.

[2] V. V. Nguyen and J. H. Lee, "Full-body imitation of human motions with Kinect and heterogeneous kinematic structure of humanoid robot," IEEE/SICE International Symposium on System Integration (SII), 2012, pp. 93-98.

[3] L. Dongheui, C. Ott, Y. Nakamura, and G. Hirzinger, "Physical human robot interaction in imitation learning," IEEE International Conference on Robotics and Automation (ICRA), 2011, pp. 3439-3440.

[4] T. Petric and L. Zlajpah, "Smooth transition between tasks on a kinematic control level: Application to self collision avoidance for two Kuka LWR robots," IEEE International Conference on Robotics and Biomimetics (ROBIO), 2011, pp. 162-167.

[5] C. Ott, L. Dongheui, and Y. Nakamura, "Motion capture based human motion recognition and imitation by direct marker control," IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2008, pp. 399-405.

[6] J. B. Cole, D. B. Grimes, and R. P. N. Rao, "Learning full-body motions from monocular vision: dynamic imitation in a humanoid robot," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2007, pp. 240-246.

[7] G. Pons-Moll, A. Baak, T. Helten, Mu, x, M. ller, H. P. Seidel, and B. Rosenhahn, "Multisensor-fusion for 3D full-body human motion capture," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 663-670.

[8] P. Azad, T. Asfour, and R. Dillmann, "Toward an unified representation for imitation of human motion on humanoids," IEEE International Conference on Robotics and Automation (ICRA), 2007, pp. 2558-2563.

[9] M. Do, P. Azad, T. Asfour, and R. Dillmann, "Imitation of human motion on a humanoid robot using non-linear optimization," IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2008, pp. 545-552.

[10] A. P. Shon, J. J. Storz, and R. P. N. Rao, "Towards a Real-Time Bayesian Imitation System for a Humanoid Robot," IEEE International Conference on Robotics and Automation (ICRA), 2007, pp. 2847-2852.

[11] T. Kroger, A. Tomiczek, and F. M. Wahl, "Towards on-line trajectory computation," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2006, pp. 736-741.

[12] T. Kroger and J. Padial, "Simple and robust visual servo control of robot arms using an on-line trajectory generator," IEEE International Conference on Robotics and Automation (ICRA), 2012, pp. 4862-4869.

[13] R. C. Luo, Y. W. Perng, B. H. Shih, and Y. H. Tsai, “Cartesian position and force control with adaptive impedance/compliance capabilities for a humanoid robot arm,” IEEE International Conference on Robotics and Automation (ICRA), 2013.