1　学校代码：10015　　　　　　　　2　　　　　　　密级：公开

北京印刷学院

硕士学位论文

混合人形机器人通过 MR 控制

Hybrid Humanoid Controlled via MR

作者姓名：SHAMAIL HASANAIN　　　　　　学　号：2017MA072

导师姓名：Mr. Qi Yuan Sheng　导师职称：Dean

学科名称：Automation Engineering　　　学位级别：硕士

2019 年 6 月

Acknowledgments

With special thanks to my supervisor, Mr. Qi Yuan Sheng for his supervision and great support. His invaluable and continuous help of constructive suggestions and comments throughout the thesis work and research contributed to the success of this work.

Special thanks to all our friends and family and others for their moral support and kindness during the project. Thank you for the great friendship and memories.

ABSTRACT

How can we see how humanoids in the actual environment sense, planned and behaved? We propose a Mixed Reality (MR) environment to view and review internal parameters that are computed against the actual environment by the humanoids. The parameters snapped on each sampling time are treated as data streams in the environment and stored in real time on the distributed log servers. 3D graphical parameter representation helps us to observe multi-dimensional parameters in plurality. For viewing the representation, a video – view through head mounted display is used. From arbitrary viewpoints, the stored parameters can be projected onto the current actual scene. The Mixed Reality environment becomes a powerful tool for the autonomous behavior developer to debug the actual behavior as opposed to the actual environment by viewing and reviewing functions. It also includes the concept of deep learning of humanoid robots during the learning phase. This thesis describes the full-size humanoid system implementation, HRP-2 and shows some experimental examples.

我们怎样才能看到实际环境中的人形生物是如何感知，计划和行为的呢？我们提出了一个混合现实环境来查看和检查人形生物体对实际环境计算的内部参数。在每个采样时间捕捉的参数被视为环境中的数据流，并实时存储在分布式日志服务器上。 3D 图形参数表示有助于我们观察很多的多维参数。为了查看表示，使用通过头戴式显示器的视频 - 视图。从任意角度来看，存储的参数可以投影到当前的实际场景上。混合现实环境成为自主行为开发人员通过查看和查看功能来调试实际行为而不是实际环境的强大工具。它还包括学习阶段人形机器人深度学习的概念。本文描述了全尺寸人形系统的实现，并展示了一些实验例子。

KEYWORDS: Mixed Reality, Mobile First to AI First, Hybrid, Forward Kinematics, Inverse Kinematics, Humanoid Complexity, Head Mounted Display, data acquisition, data manipulation, database usage, Augmented reality.

Table of Contents

Chapter 1

1.1 Introduction

The use of robots has increased in our daily life for the past few decades and now people knows how convenient it can be to use robots for their daily life tasks. But as the time passes the improvement is occurring in the technology of robotics which gradually lead to innovation of humanoid. Humanoid looks like human in appearance. As the time passed, artificial intelligence used continued improving. So more and more intelligent robots came into being. It is the fact that no matter to what extent we make our robots artificial intelligent, it may need hundreds of years to reach the intelligence of human beings. So, for now, we can't let the robot to do all of our tasks for example factory monitoring tasks.

The same time the world where you exist without physical appearance. In other words, we can say it's an illusion of three dimensional, interactive, computer generated reality in which sound, sight and even touch is simulated to create pictures, sounds and objects that actually seems real. For that we need to wear virtual glasses, also called Head Mounted Display (HMD), which helps us to see things in three dimensions.

As we were experiencing VR, we took a further step in VR world. As we wanted this reality to be not only seen through virtual glasses but also experience it as if it would have been occurring in front of an individual. So Augmented reality came into being. The definition of Augmented Reality is "a technology that superimposes a computer-generated image on HMD (Head Mounted Display) while it is moved in different directions. All the data in the form of images are sent to HMD by system that controls HMD.

In the same way, technology in this field took further steps ahead and a new research is started regarding Mixed Reality (MR) also called hybrid reality which combined real and virtual world together. Our focus will be on MR as we want to do some task like packaging using humanoid by using MR.

So, our goal is to make a system using MR that controls the robot in to modes. One mode is it will be learning from the acts that is been done by robot via MR. for example, if we want our robot to do packaging and it doesn't know how to do it then we can teach it. MR controller will be doing a packing and at the same time robot will be learning this skill. The advantage of it will be if we want our robot to do this task independently then it will have the capability to do so. The other mode will not be regarding learning any skill as the robot will only be doing tasks as it is told by MR user. In the other task, process would be faster as the data manipulation will be very less.

1.2 Historical review of Humanoids

Built in 1973 by the university of Waseda in Japan, the first humanoid robot was called WABOT-1, which stands for WAseda RoBOT 1. Although not embedded, it is capable of walking, recognizing and manipulating some visualized objects, understanding spoken language and using an artificial voice. Twelve years later, its next version, WABOT-2, was introduced with the remarkable ability to play the piano. In 1986 Honda secretly started a project on this field and 10 years later, they presented P2 which is the first embedded humanoid robot able to stably walk followed by P3. The world famous ASIMO which is a white child-size astronaut –like humanoid robot, was presented by Honda in 2000 with an astonishingly friendly design. It is sometimes believed that the impressive design and capabilities of ASIMO triggered the research on humanoid robots worldwide. Some of them are shown in the figure 1.1

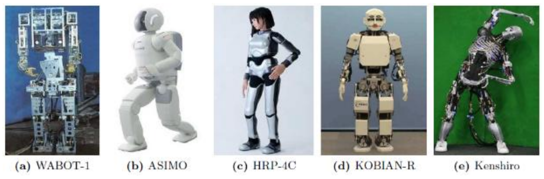
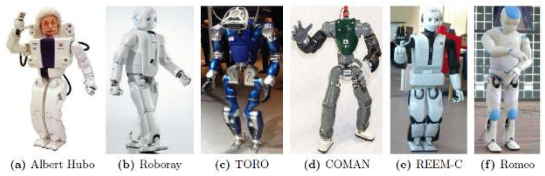


Fig 1.1

After Honda's outstanding work, Japan was the country that led the humanoid robot development. A remarkable step was the humanoid Robotics Project **HRP,** launched in 1998 by the Japanese Ministry of Economy, Trade and Industry METI and developed jointly with Kawada Industries by the Japanese National Institute of Advanced Industrial Science and Technology. HRP-2P in 2002, HRP-2 in 2004, HRP-3 in 2007, HRP-4C (Cybernetic Human), which has the head and figure of a Japanese girl, and HRP-4 in 2010 were developed within this project. In 2001, 2004 and 2005 respectively, Fujitsu Automation introduced the small humanoid robots HOAP-1, HOAP-2, and HOAP-3. Sony also developed a small humanoid robot initially called SDR-3X (Sony Dream Robot) but later renamed QRIO which was presented in 2003. Toyota released Partner in 2005, a robot capable of playing the trumpet, and in 2007 presented a version capable of playing the violin Waseda University continued humanoid development and introduced WABIAN in 1996, WABIAN-2R in 2006 and the emotional expression robot KOBIAN-R in 2008, which is capable of displaying various facial expressions. The university of Tokyo also developed a series of humanoid robots: H5, H6 in 2000, and H7 in 2006, as well as musculoskeletal robots like Henta, a tendon – driven robot, in 2003, Kotaroin 2006, its new version Kojiroin 2007 and more recently, in 2012, the Impressive Kenshiro. In 2013, the former Japanese Schaft Inc, based on HRP-2 series, presented its latest version of the Schaft robot, which is currently one of the most powerful humanoid robots.

Another Country that started developing humanoid robots early on was South Korea. The Korea Advanced institute of science and Technology developed several series of powerful humanoid robots: KHR-1 in 2002, KHR-2 in 2004, KHR-3 in 2005, Albert HUBO, Albert Einstein's android face a robot in 2006, a robot in 2005. In addition, the Korean Institute of Science and Technology and Samsung Electronics participated in the Korean Ministry of Information and Communication Network – based Humanoid Project and developed the world's first network – based Humanoids, MAHRU-II and AHRA in 2005. In 2007, MAHRU – R in 2008, MAHRU-Z in 2010 were introduced. Samsung recently introduced the 2012 torque controlled robot Robary.



You can see some of these robots in the Figure 1.2

Fig 1.2

In some European countries, humanoid robots have also been developed. In Germany, as a result of the JOHNNIE Project, the technical University of Munich developed LOLA in 2009, and in 2013 the German Aerospace Center (DLR: Deusches Zentrum fur luftund Raumfahrt) developed TORO, which was built on the basis of its two previous legs biped DLR. In cooperation with other universities and research laboratories throughout Europe, the Italian institute of Technology (IIT) developed the torque controlled iCub (Cognitive Universal Body) in 2012 as part of the RobotCub project and COMAN (COMplaint HuMANoid Platform). In Spain, Carlos III University built a humanoid called Rh-1 in 2007 and in 2011 introduced TEO. In addition, the Barcelona-based Pal Robotics company built humanoid REEM-B in 2006 as a service robotics research platform, and more recently, in 2014, presented REEM-C with a more human-friendly design. The French company Aldebaran Robotics developed Nao in 2007, the most widely spread small humanoid robot in the world and used for research in various fields. In 2014 they presented Romeo, a child size torque controlled humanoid developed as aa project between the company and several French research labs.

In the United States, in cooperation with ATR (International Advanced Telecommunications Research Institute) in Japan, the first humanoid robots were built by the SARCOS Research Corporation. The hydraulic robots Erato DB (Dynamic Brain) were Developed in 2000 and CBi (Computational Brain interface) in 2006, but they are sometimes simply called SARCOS robots. The development of American Humanoid robots has increased significantly in recent years, due in part to Robotics Challenge (DRC) of DARPA (Defense Advanced Research Projects Agency). In 2010, Virginia Tech developed CHARLI (Cognitive Humanoid Robot with Learning Intelligence) and THOR (Tactical Hazardous Operations Robot), which is expected to be completed by 2014. The National Robotics Engineering Center (NREC), part of the Robotics Institute at Carnegie Mellon University, built the 2013 DRC robot CHIMP (CMU Highly Intelligent Mobile Platform). This robot shows the peculiarity that it uses tracks for its locomotion, although it has legs, which provide a robust static balance. NASA John Space Center (JSC) developed and introduced the humanoid Valkyrie in 2013 for the DRC as well. Boston Dynamics built PETMAN in 2012 and released the hydraulic Atlas in 2013, one of the most powerful humanoid robots currently being built specifically for the DRC. Some of these robots in Fig 1.3

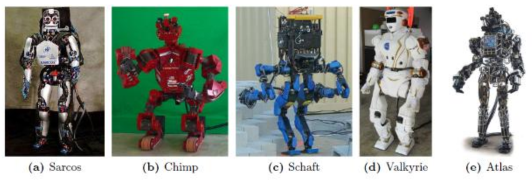


Fig 1.3

2 Chapter 2 Research Status and Development

2.1 Introduction

The world is much interested in field of Virtual Reality as it is relatively new technology and even big companies (like Microsoft, Google, Lenovo, Sony, etc) are making projects using this field. Normal use of this technology is VR glasses which is used everywhere. We can normally experience it in the form of experiencing a beautiful scene using HMD (Head Mounted Display). This is the evolution towards the new era of artificial intelligence. This development of the new generation will form information and communication industry and will shift from Mobile First to AI First, which has become commonly acknowledged by tech giants. Over the last couple of years, Google and Facebook both decided to make AI the focus of their future strategies at their global developer conferences.

The link between VR and AI lies in rendering processing, perception interaction and rendering processing aspects. In the perception interaction aspect, tech giants focus on scenario splitting, identification, locating and reconstruction based on AI. For example, Google Lens uses AI to identify contents in the pictures and Tango and the maps team use ambient 3D modeling to achieve indoor SLAM. In the rendering processing aspect, rendering requires a huge amount of computing resources and is unsatisfactory in image noises and rendering time. Rendering technologies based on deep learning can greatly improve image quality and shorten rendering time. On the other hand, VR drives AI development mainly from the universal AI aspect. AI tends to evolve from specific applications to universal applications, which depends on the accumulation of a vast amount of training data. However, it is difficult to collect such volumes of data with certain data structures in the real world. Therefore, well-known startups such as improbable and Open AI are dedicated to using virtual worlds instead of the real world for collection of training data. By building a complex and large scale virtual world with a large population, various modeling experiments can be performing, such as the prevention and control of epidemic diseases and impacts of important polices including real estate polices. In this way, a large amount of training data can be acquired.

2.2 Research

Our goal is to make a system using MR that controls the robot in two modes. One mode is it will be learning from the acts that is been done by robot via MR. For example, if we want our robot to do packaging and it does not know how to do it then we can teach it. MR controller will be doing a packing and at the same time robot will be learning this skill. The advantage of it will be if we want our robot to do this task independently then it will have the capability to do so. The other mode will not be regarding learning any skill as the robot will only be doing task as it is told by MR user. In the other task, process would be faster as the data manipulation will be very less.

2.3 Humanoid Robot Control

As the robot will be controlled with user movement and autonomous movement seen and observed using MR. So the robot's ability to move and interact with the environment is one of the main differences between a real robot and a regular desktop computer. The first part of this thesis will include algorithms designed to move the humanoid iCub. We will first introduce the basic formalization of robotic control. Moving a robot in desired ways is a difficult problem, which becomes harder the more complex the robot becomes. In our case, desired movement of robot will be done based on user movements using different methods of data transmission. We can still manually figure out a control algorithm for a simple two – wheeled robot: turn right wheels, turn left wheels. Turn right wheels, and turn both wheels to move forward. However, such simple approaches do not work for a humanoid like the iCub upper body with 41 degrees of freedom. Therefore, we need algorithms that translate a high-level description of our desire to lower level robot commands. We will introduce the formalisms used in robot cinematics and the typical approaches used in this field. Most of these approaches are fundamentally limited due to strict kinematics problem.

2.3.1 Kinematic Chain

A humanoid is constructed of rigid bodies that are connected throughout joints, controlled by electric motors, and links that are rigid connections (Fig 2.1). The joint angles change by turning the engines and the robot moves. Its angle and angular velocity describes the state of a joint. We will generally ignore the angular velocities in this thesis and let the controllers at the lower level take care of them.

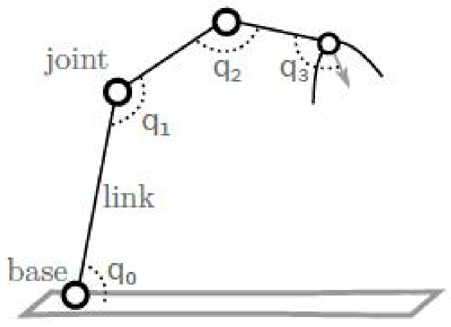


Fig 2.1

Robot Arm A robot's topology is defined by connecting a base and rigid bodies that are via joints and links. The robot moves by changing its joints angles.

Ignoring the angular velocity, we represent all common angles in a single vector $q \in Q \in \mathbb{R}^J$, called the robot's joint parameters. Here Q is the set of all possible parameters together with the number of joints. The *i*th joint is denoted by $q_i \in \mathbb{R}$, an index between 1 and j, and q_i is a real number that describes the angle.

In order to formally represent the robot's topology, we first define the robot base, typically the hip or torso, which is our reference point. From the base to each body part, we follow the skeletal structure, forming a chain. The finger connects with the hand, the hand with the arm, the arm with the torso, and the torso with the robot's hip, for example. A transformation can represent each of these connections, and these transformations can be chained together to calculate the location and orientation of each part of the body. The typically used body part of for manipulation, such as a gripper and a humanoid's hand, is called an end effector.

The resulting chains are represented by a tree structure called the kinematic chain. This calculation is called forward kinematics due to a set of joint parameters and the kinematics chain the positions and orientations of all body parts can be calculated.

2.3.2 Forward Kinematics

We use transformation matrices to represent the transformations resulting from the cinematics chain to calculate the robot's cinematic configuration.

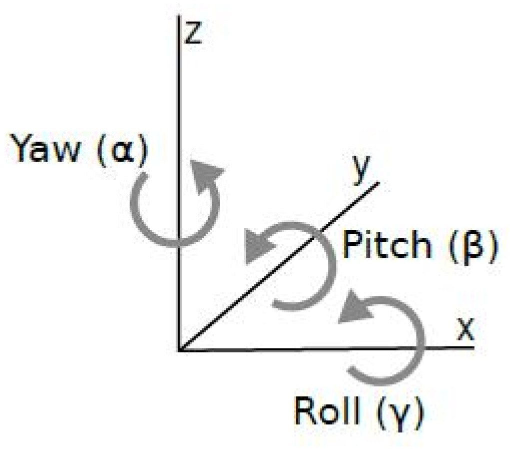


Fig 2.2

For defining a Rotation Matrix, it is necessary to consider the rotations over all three axes. The x-axis rotation is called the roll, the pitch is called over the y-axis, and the yaw is called the z-axis.

Transformations defines rotations, e.g., to represent a joint rotation, and translations, e.g., to represent a connection between two joints. In Cartesian space, given a vector v ∈ $\mathbb{R}^3$, for the calculation of the transformed vector $v_{transformed}$ by matrix multiplication, the transformation matrix R ∈ $\mathbb{R}^{3x3}$ can be used.

$$V = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad (2.1)$$

$$V_{transformed} = R.v \qquad (2.2)$$

Rotation: if we define x, y and z-axis, a rotation is called a roll over the x-axis, the y-axis is called a pitch and the z-axis is called yaw.

$$R_{yaw}(\alpha) = \begin{bmatrix} cos \propto & -\sin \propto & 0 \\ \sin \propto & \cos \propto & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (2.3)$$

$$R_{pitch}(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \qquad (2.4)$$

$$R_{roll}(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & sin & \cos\gamma \end{bmatrix} \quad (2.5)$$

Multiplying these matrices can create a general transformation matrix:

$$R_{yaw,pitch,roll}(\alpha, \beta, \gamma) = R_{yaw}(\alpha) \;.\; R_{pitch}(\beta) \;.\; R_{roll}(\gamma) \quad (2.6)$$

Translations: we represent the positions with a vector that holds the position x, y, and z and a fourth value 1:

$$V = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (2.7)$$

This allows us to combine translations defined by a vector t = [t_x, t_y, t_z] ∈ R^3, with rotations in a single matrix 4 as follows:

$$R(\alpha,\beta,\gamma,1) = \begin{bmatrix} R(\propto, \beta, \gamma) & \begin{matrix} t_x \\ t_y \\ t_z \end{matrix} \\ 0\;\;0\;\;0 & 1 \end{bmatrix} \qquad (2.8)$$

This final matrix of transformation is parameterized by the parameters α,β,γ,t. note that if all parameters are 0, the matrix becomes an identity matrix. These transformations can represent all joints and links in the iCub. The parameters depend on their angle q_i for joints, while links are stiff and have a fixed parameterization. Generally speaking, we write R(q) to show that a certain matrix of transformation depends on the parameters of the joint.

Since matrix multiplications are linear, multiplying all matrices can effectively calculate the sequences of transformations in the kinematic chain. For example, from the base position the position of the hand is calculated as follows;

$$V_{hand} = R_{lower\ arm \rightarrow hand}(q) \cdot R_{upperarm \rightarrow lower\ arm}(q) \cdot R_{torso \rightarrow upperarm}(q) \cdot R_{base \rightarrow torso}(q) \cdot v_{base} = R_{base \rightarrow hand} \cdot v_{base} \qquad (2.9)$$

Orientation: we define a vector pointing in a certain direction with regard to the relevant body part and calculate the final orientation to calculate the orientation of a body part, for example the hand. We might be interested, for example, in a vector coming out of the hand's palm:

$$\mathbf{V_{palm\_dir} = v_{palm\_point} - v_{hand}} \qquad \mathbf{(2.10)}$$

Where v_{palm} point is calculated by adding translation:

$$\mathbf{V_{palm\text{-}point} = R_{hand \rightarrow palm\_point} \cdot R_{base \rightarrow hand} \cdot v_{base}} \qquad \mathbf{(2.11)}$$

To calculate such directional vectors quickly, the complete Rbase manual transformation matrix is stored during the calculation of forward kinematics. Position v and orientation r of all body parts can now be calculated efficiently. Generally speaking, we call the set of all body parts positions and orientations the kinematic configuration p:

$$\mathbf{P = \{(v_{lefthand}, r_{lefthand}), (v_{righthand}, r_{righthand}), \ldots\} \in p}$$

To simplify notation, we use function f, called forward kinematics, to describe the process of calculating the kinematics configuration p from the parameters q described above:

$$f(q) = p \qquad (2.12)$$

Software: we are applying the MoBeE framework for forward cinematics in this thesis. It takes an XML configuration file that defines the robot topology including all joints and uses the SOLID 3.5.6 [libSolid: Collision Detection Library] collision detection library to perform fast forward kinematics and collision detection can be added to model the environment also visualizing the resulting cinematics configuration scan.

2.3.3 Inverse Kinematics

As discussed in view of the joint parameters q, the forward kinematics function f calculates the kinematics configuration p.

$$f(q) = p \qquad (2.13)$$

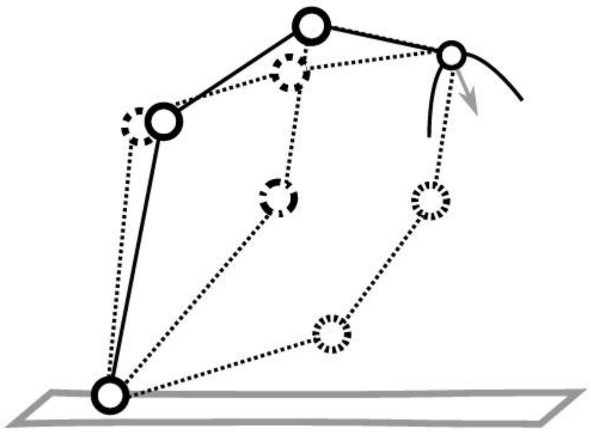


Fig 2.3

Inverse kinetics: IK is trying to find common parameters that fit a desired configuration of kinematics. This is a difficult problem as there are typically many solutions or even infinite many. IK's main goal is to find stable methods to deal with these problems

But to control the robot we need the reverse: we want to find the corresponding joint parameters due to the desired configuration of body parts. This issue is called inverse kinematics

$$f^{-1} (p) = q \qquad (2.14)$$

where f^{-1} is the opposite of the function of forward kinematics, and p is (a subset of) a kinematic configuration. Although it is easy to calculate, its inverse f^{-1} cannot normally be directly calculated and may not be a well - defined function. First, it is not known if the desired configuration 'p' even has a solution. And if there are multiple (if not infinite) solutions typically exists. Figure 2.3 shows several solutions for a two-dimensional arm in which the gripper is in the desired and orientated position. There are already infinite solutions to this problem with only a few degrees of freedom in a two-dimensional world. Just as we go into three dimensional spaces and many degrees of freedom, this problem gets bigger.

One solution is to limit the problem in such a way that there is only one defined solution and that f^{-1} can be expressed analytically. This is very difficult due to the non-linearity and size of the joint parameter space and is only realistic in very restricted settings.

2.3.4 Jacobian Based Approach

A more popular approach is to give up a global f^{-1} analytical expression and resort to iterative local optimization. We can define a positive real established error function over a cinematic configuration p which decreases as we approach our desired configuration p.

$$E(f(q), p^r) \geq 0$$

$$q^* = \min_q E(f(q), p^r)$$

Then we can find the optimal joint parameters that minimize error E by calculating the gradient to the joint parameters and sue it to optimize.

$$E\ (f\,(q), p^r\,)$$

A simple way would be to follow the gradient's negative iteratively until function E no longer decreases:

$$q_{new} = q - \nabla_q E\,(f\,(q), p^r\,)$$

Jacobian

The derivative must be calculated through all transformation matrices in the cinematic chain to calculate the gradient of the error function E towards the joint parameters. Since these matrices have vector – valued outputs, there is no regular gradient; instead, it is a matrix formed for all combinations of elements in the output and input vector by all first – order partial derivatives. This matrix is called the Jacobian Matrix. Control methods using the gradient are said to be controllers based on Jacobian or gradient.

However, there are several problems with these Jacobian approaches. Since the search space is highly nonlinear and the gradient is only a local linearization of this search space, sub – optional solutions can be achieved simply by following the gradient. Typical examples are local minima where the solver is stuck in the search space in a local dip, or plateaus where the gradient is very small, causing many iterations to be wasted by the solver.

Singularities, search space areas where the gradient misconducts, are another major problem. Such problems can occur when the desired kinematics configuration is not solved, e.g. when a goal is out of reach. Or when search space regularizations were added to help the solver interact in complex ways, causing the search space to be undefined.

Another example is shown in fig 2.4, where a robotic arm is fully stretched and the objective is to move the arm down (referred to as the red arrow). The gradient does not provide a solution here because every change in the joint angles leads to a sideways movement that is orthogonal to the desired objective. The gradient will result in the right movement only after the arm is slightly bent. When the complexity of the robot increase, these problems are increasingly present and are especially true for a humanoid robot.

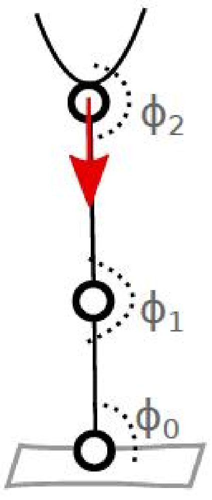


Fig 2.4 Singularities are function space points where the function has an abnormal behavior and are a major problem for IK. In this example, an arm stretched upwards and now has to move back down. However, the gradient around this kinematic configuration is zero in this stretched position, as each local joint movement results in the gripper's horizontal movement: not vertical. For gradient based controllers, such points are problematic.

Nevertheless, approaches based on Jacobian are still very popular. For example, Jacobian control was applied to humanoids by Baerlocher and Boulic, 2004. However, the proper setting of restrictions is still a process involved and the search space must be reduced in order to keep the degrees of freedom under control.

2.4 Control

We distinguish dynamic control from kinematic control. With dynamic control, we refer to control algorithms that directly calculate the robot's motor forces, taking into account the robot's dynamics. With kinematic control, we refer to algorithms resulting in static kinematic configurations used as targets for another program calculating the forces. This thesis concerns only kinematics control dictated by the VR system, allowing us to focus on the motion's kinematics form and ignore the dynamic aspect that would require the robot's physical models to be much more detailed. It remains for future work to extend the methods in this thesis to dynamic control.

2.5 State of Art

From direct solvers to iterated optimization and online learning. The traditional approach to IK is to find explicitly by limiting the problem until a closed – form solution is reached. This approach is very fast but in the constraints that can be used it requires careful engineering and is restrictive. Numerical approaches find iterative optimization using the gradient f-1 indirectly, often by calculating the pseudo – inverse or Jacobian transposition. Recent work applies such methods to humanoid robots and adds an effective way to handle hard constraints prioritized. However, the constraints must be carefully set up and the systems dimensionality must be controlled to effectively calculate the reverse. Furthermore, Jacobian methods look only at the local slope and do not take into consideration the curvature and linearity.

Current robotic control systems are difficult to set up, relying on strict kinematics formulations, limiting the restrictions that can be used to design movements. Sampling methods have been used to allow greater freedom in the constraints. These methods only use forward model evaluation, alleviating constraints such as continuity and smoothness on f, but they do well to high dimensional kinematic models.

2.6 Concluding Remarks

Controlling a robot, as we have seen, is a complex issue and an ongoing research field. Most field algorithms solve the problem by restricting it until it can be analytically solved or using Jacobian based approaches. Relatively new in the field are sampling based approaches that use more computing power but allow much more freedom to define the robot’s movements. We will continue in this direction and introduce sampling based algorithms that enable flexible control of humanoid robots. The increased use of computational power is offset by the method’s flexibility. We develop ways of reusing previous calculations, parallelizing the algorithm and providing an online version in addition to mitigate the computational burden.

3 Chapter 3 Content and Methodology

This case study aims to emphasize that the described framework for movement generation, used to make humanoid robots perform complex tasks, provides a powerful tool for inferring human movement organization. In fact, complex inhuman whole body movements rely on a set of motor goals and constraints that are not easily identifiable. In particular, it is a difficult question to determine how the motor control is distributed simultaneously to the guarantee, the postural balance and the displacements of the limbs required for the task. The proposed framework for motion generation provides a standard way for synthesizing such complex movements as an ordered set of goals and constraints. It is then possible to test some structural hypotheses about the organization of the motor control associated with a given task using the human body's a dynamic model. By comparing the variation of key parameters given by the motion generation software with those measured on human subjects performing the same task, it is possible to evaluate the hypotheses proposed and go back and forth from observation to simulation, the key elements about the dynamic structure of human movements can then be inferred.

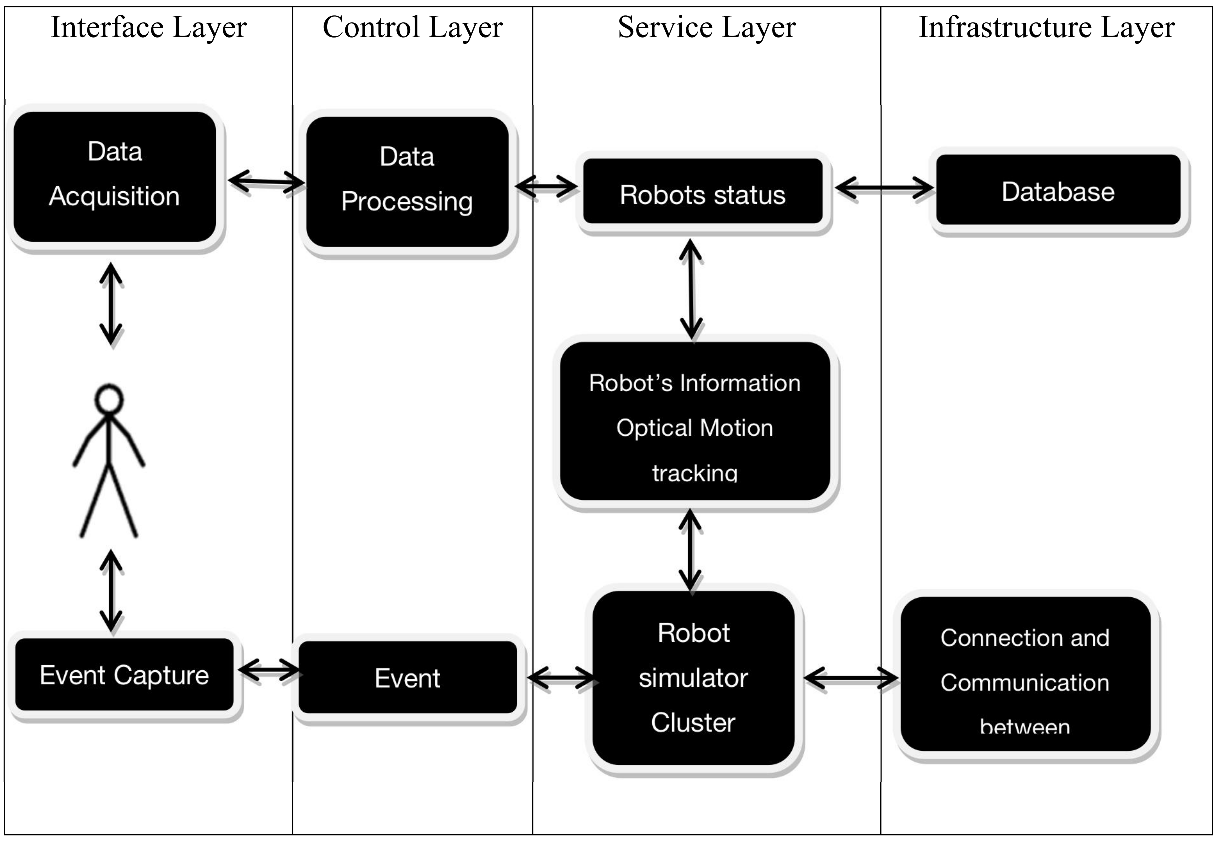


Table 1

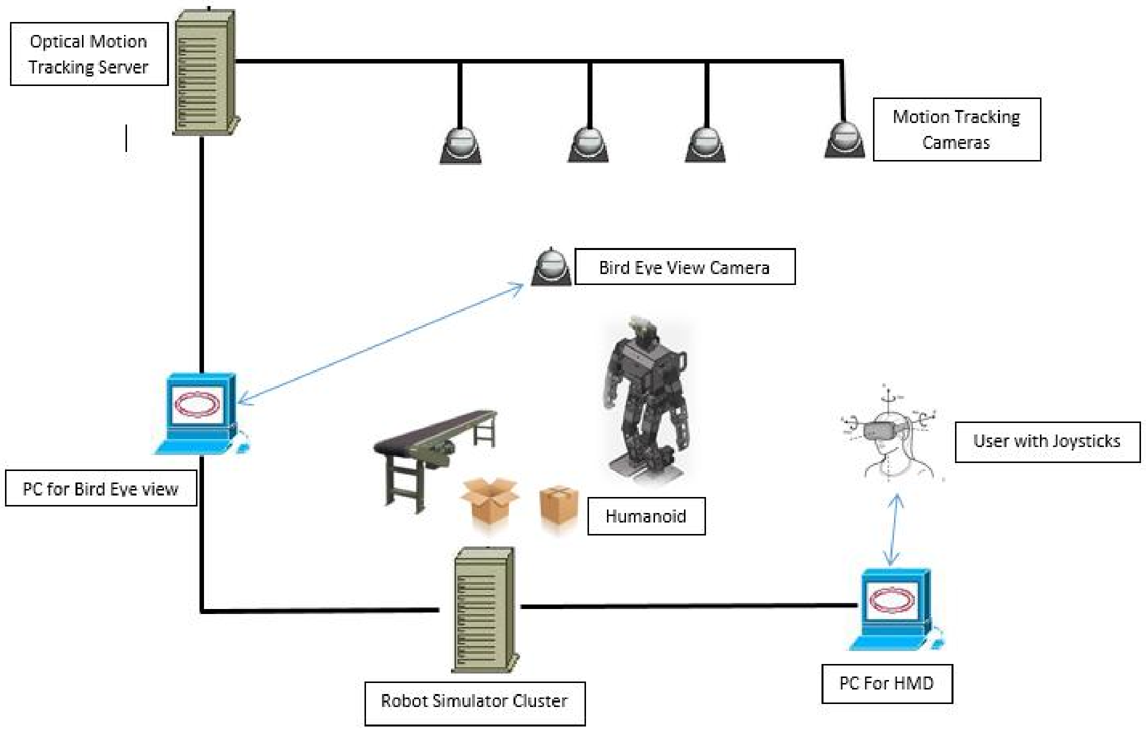

Optical Motion Tracking Server
Motion Tracking Cameras
Bird Eye View Camera
User with Joysticks
PC for Bird Eye view
Humanoid
Robot Simulator Cluster
PC For HMD

Fig 3.1

4 Chapter 4 Significance of study

4.1 Environment

A robot must be able to see, act, and react continuously to be useful. Perception is a key requirement to customize robot motion to the environment, enabling more successful, autonomous interactions. The first important step in this direction is to understand the robot’s environment. Coming back to the playing chess example, this would compare to finding the chess board and each of the chess piece or even just realizing that the scene has a chess board and pieces.

Much research in psychology, cognitive science, neuroscience and biology focuses on vision and the visual system. A major problem with perception is that what people see is not merely a simple translation of input stimuli (comparative illusions). The development of artificial vision systems is an important area of research to build robots that can understand their surroundings.

Computer Vision sometimes referred to as Robot Vision when applied in a robotic system – generally describes the field of research in which images are acquired processed, analyzed and understood in order to make observation based decisions. Computer vision and AI fields were close at the beginning, but over the past decades they have gone separately. A trend is emerging to reunite these fields.

While the areas of computer vision and image processing are not clearly defined the latter is commonly seen as referring to a subsection of computer vision. Image processing techniques generally provide ways to extract information from image data and can be grouped into the following categories: preprocessing, extraction of features, segmentation and high level processing in computer vision another important topic is image understanding. the goal is to imitate the abilities of the human(visual) perception system with the help of geometry, physics, statics and learning.

Vision research is referred to as robot vision or machine vision for the special requirements of robotic systems. For example, visual feedback has been widely used for obstacle avoidance, mapping and localization in mobile robot applications. With the advancement of humanoids and increased interest in working around humans, object detection and manipulation are increasingly driving the development of robot vision systems. A major problem is whether or not the image data contains some specific object, feature, or activity. While this has been researched for quite some time now, the general case of detecting arbitrary objects in arbitrary situations. From a robot vision point of view, this means that the robot is required to detect previously unknown objects in its environment and be able to build models to memorize and identify them in the future. Most of the work relies heavily on artificial landmarks and fiduciary markers to simply the problem of detection. In addition, existing methods can best solve this problem for specific objects and in specific situations. The reader is referred to the excellent survey by Kragic and Vincze for a detailed introduction and overview of the foundations and current trends.

4.2 Environment Boundaries

Over the past decades, computer vision has become an increasingly prominent research topic, including the field of robotics. Robots can interact with the world around them like humans and animals. While most robot vision research uses only passive observations to focus on understanding the world, these interactions with the environment provide valuable information for building better visual systems. Connecting manipulation command with visual input allows a robot to create methods to actively explore its environment. These connections exist within the human brain between motor actions and observations and are an important aspect of human development.

Only after observing the scene and having an idea of which objects are in the environment can the robot begin to interact with them in a safe fashion. In the example of chess, even if the board’s condition and where it is located is known moving a certain piece of chess from one field to another without toppling other pieces is still a difficult problem on its own. In fact, even at a very young age, children have significantly better hand movements than any humanoid robots currently available. But it’s not trivial to manipulate arbitrary objects, even for humans. The development of manual control in children is not matured until the age of 8 – 10 years for an apparently simple, prototypical tsk of grasping precision. In addition, completely is staggeringly high, as can be seen by the number of neurons comprising arm and hand control. Even after learning manipulation skills, they are constantly adapted to yield desired results through a loop of perception action. In infants, different specializations in visual pathways may develop information relevant for visual cognition to be extracted and encoded, as well as information about the location and capability of objects. This suggests that vision and action in the human brain are very closely integrated.

The robotic grasping of objects has made good progress in recent years. The various manipulators, mainly hands and grippers, and techniques have improved significantly. There have also been designed novel grippers concepts and some are quite ingenious solutions to a number of problems. One such example is the granular gripper manufactured by Brown et al, which consists of grounded coffee beans that can flow around the object and then fasten in position by creating a vacuum. Recently, this concept has been extended to a full size arm in the elephant trunk style. Recent results also highlight the advanced state of grasping research in terms of how to grasp objects with regular grippers and hands. For example, with their research, Martin Shepard et al. showed that robots can pick up non – rigid objects like towels. Their robot can reliability and robustly pick up a randomly dropped towel from a table by going through a sequence of partially on the table, partially in the air, vision based re-grasps and manipulations. In the DARPA ARM project, which aims to create highly autonomous manipulators capable of serving multiple purposes across a wide variety of applications, NASA's JPL winning team showed an end to end system that allows the robot to grasp various objects power drill, keys, screwdrivers, etc from a table. On the other hand, Saxena et for a robot to learn from only a small number of real world examples where there are good grasping points on a wide range of previously unknown objects.

All this let to Dr. Pratt, DARPA's manager of research related to robotics to his somewhat controversial statement Grasping is resolved at IROS in 2012. While this may be a little too optimistic, the research seems to be in a good enough state to have better integration of the system. To enable robust object manipulation, the direct interface between different components, which makes robotics such a hard but interesting field, clearly needs to be improved. Only a fully functional pick and place capable system will appear by combining sensing and control of the entire robotic platform. In order to enable a variety of objects to be picked up from different positions, the robot needs to see, act and react in a control system in which these elements are closely integrated Robotics. Vision and Control must be read for robotics operators. It puts in the spotlight the integration of these three components. It also describes common pitfalls and problems arising from integration.

4.3 Connectivity of VR and Humanoid

4.3.1 Introduction

The DARPA Robotics Challenge (DRC) was announced in October 2012, a competition in which teams would complete with the overall objective of developing semi-autonomous humanoid robots capable of performing complex tasks in hazardous, degraded, man-made environment. There were three rounds of the competition: The Virtual Robotics Challenge (VRC), the DRC Trials, and the DRC Finals. Using physical, typically humanoid robots, disaster response tasks such as driving and existing a modified Polaris, opening and walking through a door, operating a drill, and walking through debris were completed during the trials and finals. Tasks were performed in individual runs during the Trials, whereas all tasks were performed in a combined run succession by the Finals. The successful trial teams went on to the finals.

An analysis of the human robot interaction (HRI) techniques used in the DRC Trials and the DRC Finals was carried out. Both studies examined interaction design team approaches such as the number of operators, control types, how information was transmitted to the operator and different approaches to semi-autonomous task performance. The various teams used a number of strategies, but they rely predominantly on a combination of interfaces between mouse, keyboard, and game pad. The study teams averaged 5.2 active screens, 6.2 camera views and 2.5 different point cloud visualizations for the visualization of sensor data in the DRC Finals. There was an average of 1.6 active operators and 2.8 passive operators. All this represents a large amount of manpower with the aim of enabling the operator to gain proper situational awareness and task awareness of the remote environment by interpreting sensor data from a 2D interface and building a 3D mental model.

Another interesting and common feature was that teams often had different paradigms of visualization and control depending on their task. For example, one team went so far as to deliberately lock the operator's interface depending on the selected task, preventing them from switching to various visualization or control modes. This strategy is also evident from the analysis of the DRC HRI; most teams would adjust what sensor data was displayed, what control types to give the operator, and whether a gamepad or keyboard and mouse would be used. All this seems to imply that each team has spent a significant amount of effort creating a specialized interface for each task in order to provide sufficient awareness of the situation to enable the operators to complete the task within the allocated time.

Teams using the Oculus Rift Developer Kit (DK) for viewing point clouds were the only known uses of VR in the competition, but this was used as a situation awareness complement to viewing point clouds by traditional means. No teams used VR for either Visualization or control as their primary interface. A discussion with a member of one of the teams revealed they had investigated using the Oculus Rift DK, but they found it to of limited use due to its limitations and did not end up using it at the finals.

The Oculus Rift DK required the wearer to be in a fixed sitting position, ideally holding their head in a set position. While the headset could handle rotation in the pitch and yaw axis, lateral movements could be handled. If the operator significantly moved their head, the virtual world would be unable to adapt and the operator would frequently get movement sickness. Even in the ideal position when it was used, low pixel density and low refresh rate made it difficult for VR to reach its potential. Shortly after the end of DRC Finals in June 2015, several powerful consumer virtual reality headsets were released, including Oculus Rift and the HTC Vive ready for consumers. These two headsets were significant improvements over the Oculus Rift DK and while we are going to focus on the HTC Vive headset, they're largely interchangeable.

The newer headsets remove the limitation of the fixed 3D position and have accurate position tracking that allows the so-called Room Scale or VR experience ability that allows users to walk freely around a play area with their real-life movement reflected in the VR environment. Together with other improvements such as increased screen resolution and increased refresh rate, we believe that another look at using VR to visualize and control humanoid robots is warranted and that a humanoid robot VR interface has been designed and created for use in HR examinations.

4.3.2 System

While fully autonomous robots are popular in research and have succeeded in very specific fields, it is still often preferred for many complex tasks to have a human operator in the loop. Many of the teams used very little autonomy specifically for the DRC. It was typically used by the teams that had some autonomy in very specific and limited cases. For example, IHMC began with scripts that were able to perform tasks on their own and broke these scripts into smaller pieces over time. The end result was a series of steps in which the robot would plan a smaller action, and before approving the plan, the operator would either confirm the plan or more often, make adjustments to the state or plan of the robot. After the DRC Finals, many teams published papers on their robot control interfaces and learned lessons, including Team IHMC, which enabled their robot operator to command the robot through interaction with the three dimensional 3D graphic tool instead of directly specifying robot movement or joint positions. With all this in mind, we are primarily interested in an interface where most of the robot functions are retained by the operator using VR. Similar strategies were common among several teams, although they were all portrayed on 2D screen interfaces, and were usually interacted by a mouse and keyboard.

While the teams competing in the DRC had a specific goal, namely to complete a series of structured tasks in a controlled competition, it still represents one of HRI's most significant humanoid robots events and so we built on the lessons learned when building our own VR interface. For example, teams with greater success in performing tasks relied in a common reference frame on a fused display of the 3D robot avatar, point cloud and camera images. 2 interaction methods, such as a mouse and keyboard, have been used to maneuver displays and issue commands. HRI for humanoid robots inherently consists of 3D data, for which VR offers a unified solution for both 3D display and control. Our goal was to create an interface that would leverage VR to increase the operator's situation and sensitivity when viewing a remote location, while also providing adequate control methods to complete with traditional 2D mouse and keyboard interfaces screens.

Our proposed interface uses the HTC Vive a commercially available VR headset, optionally combined with the Manus VR gloves, a pair of gloves enabling accurate finger tracking as well as tracking the location of the wrists with the HTC Vive Trackers. Alternatively, to the glove, we also use the standard headset controllers.

4.3.3 Robot platform

While our interface was designed to be applicable to any similar humanoid robot, we considered NASA's Valkyrie R5. Valkyrie is a 6-foot, 200-pound humanoid robot with four hands attached in 7DOF arms. The robot consists of several sensors including a Carnegie Robotics Multi-sense, located in its head, capable of generating high and low density point clouds, depending on bandwidth requirements, as well as a stereo camera view. The root also has in the lower torso an additional pair of stereo cameras. Last but not least, the robot embedded tactile sensors in the fingers and palm to detect grasping success in addition to accurate joint state and torque tracking on each joint. The types of actions that the robot performs are moving a foot to a 3D space location and holding without taking a step; moving a foot to a 3D space location and taking a step; commanding the torso, pelvis or neck to a commanded position; and commanding one of the arms to 3D space location.

4.3.4 Software Development

Using ROS, a communication middle – ware commonly used in robotics, Valkyrie's sensors and controller interface communicate. When this project was started, the HTC Vive Linux driver lacked many important features, so we built our application within the windows OS. To communicate with the Robot, we used ROS.NET's Unity plugin that handled all communication and conversion between standard ROS topics and types into things that might be used within unity. With the settled infrastructure, we were able to use the most supported and complete feature SDK for the HTC Vive, with the added bonus of being able to leverage several additional features within unity.

4.3.5 Interface

We build our VR interface trying to take full advantage of the features available. With the operator wearing the VR headset and either wearing the gloves or controllers, the operator finds itself in a virtual world. In order to move an avatar in the virtual world, the operator can physically move around a clear world space. Since the physical workspace can be significantly smaller than the remote robot workspace, we also allowed the operator to tele port around the virtual world by pointing their controller to a location on the ground and pressing a trigger button. This allows the operator to continue navigating in remote environments that are much larger than the operator's own environment.

The robot can track its own relative movement in the world accurately through an IMU, as well as the joint state of watch of its limbs. This means that we can render a virtual robot version for the operator that updates in real time as shown in figure 4.1 with a known robot model. Using room scale, the operator can then navigate around to view the virtual robot from any angle and position.

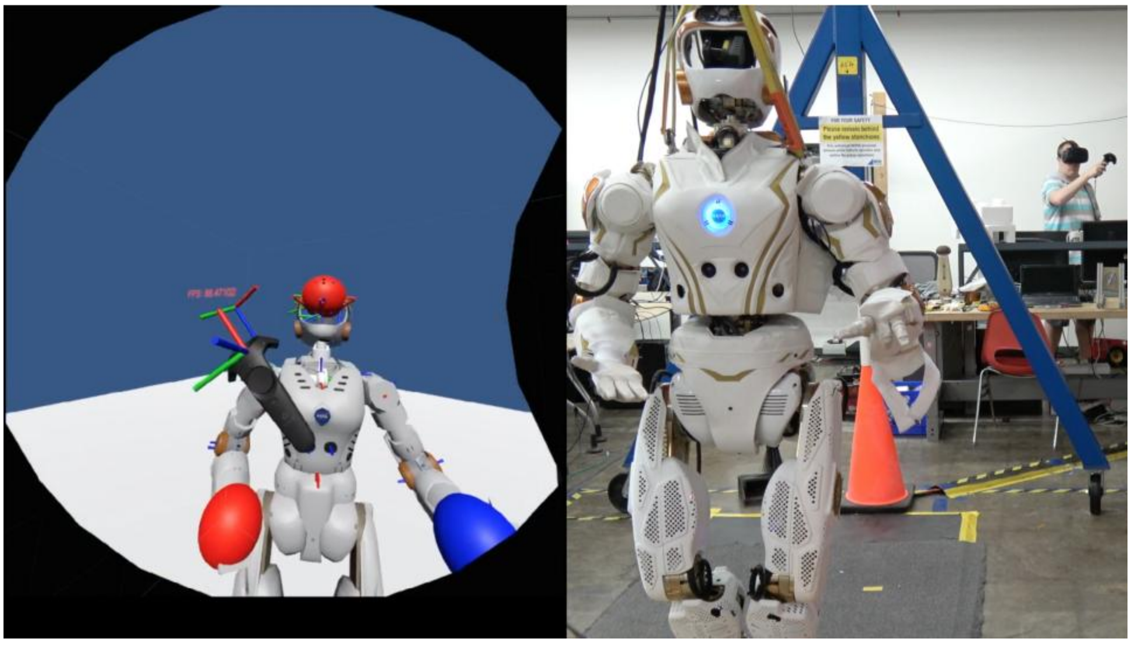


Fig 4.1

4.3.6 Rendering

Cyber sickness remains a concern, as discussed in Davis et al. while the frame rate of the system can be considered insignificant in many traditional interfaces, a frame rate above 90 frames per second is recommended for VR. One way to reduce system load is to throttle the sensor data that can work for static environments, but the operator needs up to date information in real time for dynamic environments. To this end, special care was taken when handling point clouds to ensure that it was both fast and efficient to receive and render the point cloud in VR Fig 4.2 provides an example of a cluttered table with minimal impact on system performance rendering a high density point cloud

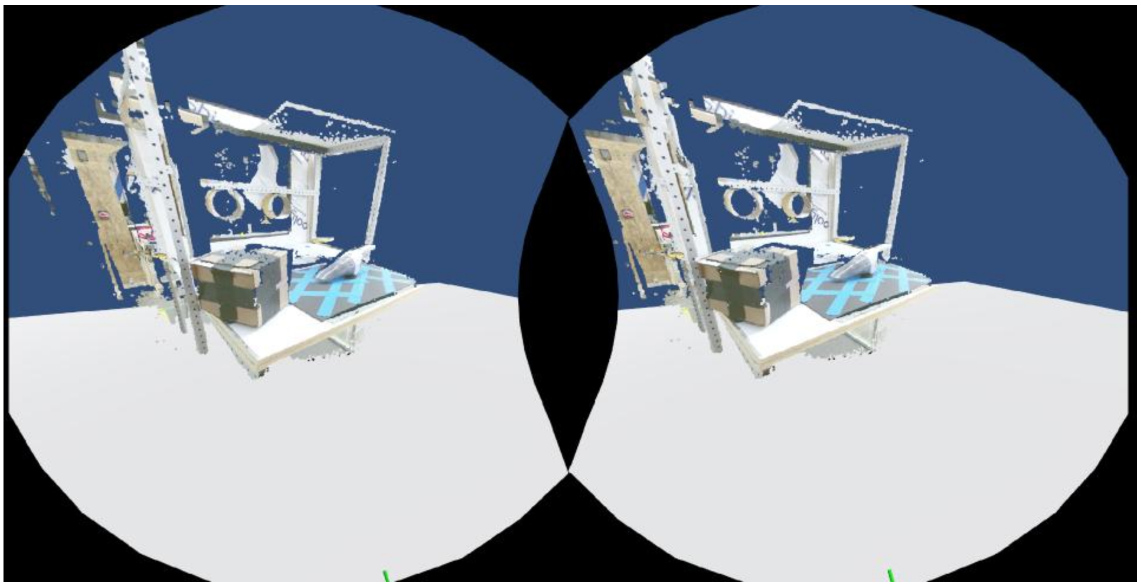


Fig 4.2

Hen rendering 3D data, VR poses several unique challenges. Unlike a 2D interface, the computer cannot simply pause rendering to load new sensor data into the graphics card. One has to worry about both the average frame rate which can be slowed down simply by the amount of data in the scene, and the maximum frame latency caused by loading new sensor data.　In an attempt to keep the total data as low as possible, it was decided to render each point as octahedrons in a point cloud, which has only 8 triangles and 6 vertices, compared to a cube with 12 triangles and 8 vertices. Voxel filtering and reconstruction of the surface are also highly effective methods to reduce the total number of points to be rendered.

Equally challenging, if not more, is the problem of reducing the maximum frame latency. When Unity receives a new point cloud. It needs to be broken into vertex, triangle and color arrays that can be understood by unity mesh objects. This can be handled on a thread other than the main rendering thread, but loading those arrays into a Unity mesh object that needs to stop rendering while loading them. As long as the user maintains an average frame rate of 90 frames per second, generally a brief drop to 50 or 60 for a single frame is not a problem, but that still only allows up to 20milli seconds to load new point cloud data into the mesh object and then into graphics card. The point cloud can be broken down into several different meshes to overcome the limitation, so that each can be loaded in less than 20 milliseconds. these meshes are divided into various groups of colors, all of which are updated in the same frame. This grouping of colors prevents the point cloud update form having discrete chunks in different frames. If there are several different point clouds from different sensors in the scene, special care is also taken to avoid updating them in the same frame to prevent a drastic increase in frame latency.

4.3.7 Usability

The operator's ability to view the point cloud in relation to the robot as shown in fig 4.3 is of particular use. Notice a doorway's low-density point cloud before the robot. Because the scene is recreated on a one to one scale for both the robot and the virtual avatar of the operator, the operator can see both the robot's position and size of the door and themselves, then maneuver their point of view throughout the virtual space as needed. This should allow operators to quickly and reliably plan their actions. This design follows recommendations from Nielsen, Goodrich, and Ricks and Okura et al, where it is suggested to use a common frame of reference with variable perspectives, environment for 3D displays and robot avatar to increase operator awareness of the environment.

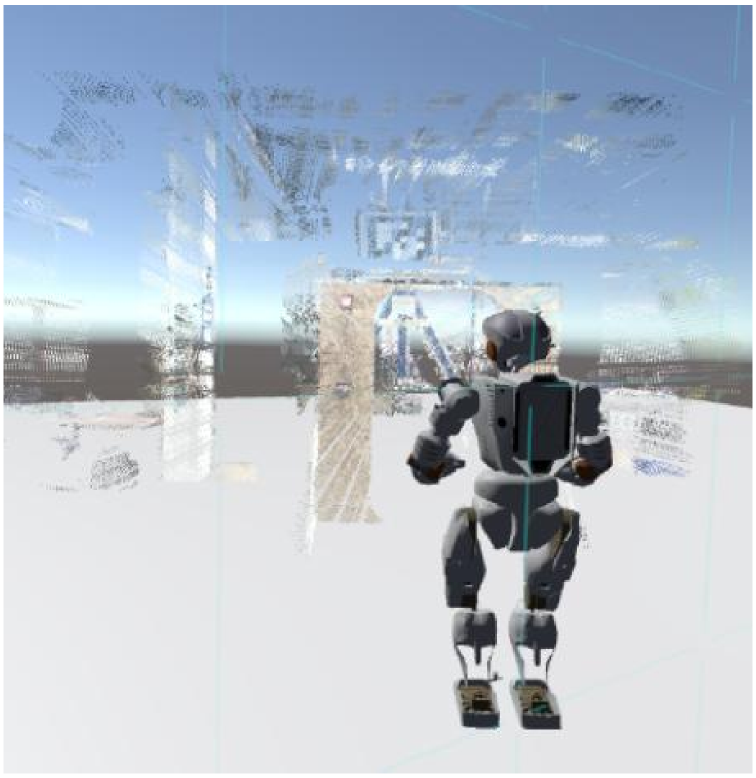


Fig 4.3

This VR system also has the ability to load any environment’s static models or even static point cloud scans. The operator can more easily locate them by adding this additional information and understand the position and environment of the robot. Static models and point clouds help fill in information about what is not currently being looked at by the robot.

While point clouds transition naturally to 3D environment, taking advantage of VR cameras poses an interesting challenge. The camera view itself is 2D and thus does not gain immediate benefit by being in a 3D space; however, when viewing the camera streams, we were able to provide ease of use. By creating rectangular windows as shown in figure 4.4, were the cameras are rendered in the virtual world. The operator can use their controller to grab the virtual windows. They can move it to a different location or resize the window once they interact with the virtual item. This allows the operator to position the windows in strategic locations: for example, where the actual camera field of view would sling the camera windows roughly. Alternatively, the operator can pin a camera window to hover from the operator at a specific location, serving as a display heads up. As the operator moves across the virtual environment, they can see the stream of camera whenever they want. These options allow the operator to retain at least the same functionality as the 2D display while offering more context and convenience. By allowing both fixed and variable perspectives, we follow the design guidelines for sensor fusion techniques shown to be successful in the DRC Finals.

4.4 Making Humanoid AI

4.4.1 Introduction

Jorge Luis Borges writes about a library of indeterminate and infinite number of books in his ‘The library of Babel’. Most of them contain nonsense, and rational relationships are almost miraculous exceptions, but there are books somewhere that contain all the universe knowledge.

A traveler who wants to find something rational in such a library have a very difficult task, as most books are only a cluster of incomprehensible letters. How could we help this traveler in the endless space of options to find a masterpiece? We need a tool that would effectively fly in this space and we’d find something inspiring thanks to it. An evolutionary algorithm could be one such tool.

The artificial Intelligence discipline constructs computational systems inspired by different aspects of life. An evolutionary algorithm is an algorithm of optimization that uses mechanisms inspired by biological evolution from Darwinian theory. AI researchers are increasingly trying to make robots dance. We used an evolutionary algorithm form called interactive Evolution to design a humanoid robots Robotic Dance system. User interactively cooperates with the system to design their own choreography of robotic dance. The interaction between humanoid robots and individuals is the key factor in their success, as they must exist in a human environment.

4.4.2 Humanoid Robotics Research

According to Brooks, Artificial Intelligence’s implicit dream was to build intelligence at the human level. For AI and robotic workers, building a humanoid is the challenge par excellence. Recent generations of humanoid robots (HR) are increasingly resembling human in shape and capacity.

Humanoid robots are robots based on the appearance of the human body at least loosely, by humanoid we mean not only a robot that has the physical appearance of human, but also a fairly advanced artificial intelligence, enabling it to reasonably approximate human behavior and interactions, talking about humanoids in the AI community.

HR’s main objective is to insert humanoid robots into the human environment new research trends consider the ability of the robot to interact safely and naturally with people. The current topics in HR are so called service and social robotics – where the goal is to get robots closer to individuals and their social needs. There are many attempts to create robots in a natural language that can express emotions or communicate. The following research topics in human robot interaction include:

Friendly human robot interfaces

Safe human robot interaction

Emotion expression and perception

Social learning

The wide spectrum of interactive robotics applications demonstrates the importance of how we work and cooperate with robots, such as assisting in health care, rehabilitation, and therapy, serving as tour guides, office assistants, and household staff. We will be engaged by social robots, entertain us, and enlighten us. Thus, while continuing to enhance autonomous capabilities is important we must not neglect to improve the relationship between human and robot.

4.4.3 Use of Artificial Intelligence Methods in Humanoid Systems

Artificial Intelligence AI attempts to make programs or computers do things that are described as indicating intelligence when performed by people. AI's goal was characterized as both building useful smart systems and understanding human intelligence. There have been thoughts of building truly smart autonomous robots since the earliest days of AI. Work in robotics has influenced work in AI and vice Versa in academic research circles. The development of intelligent algorithms has influenced neural networks, fuzzy logic, and evolutionary computation. Tolerance of uncertainties and other characteristics is applied in many advanced control applications in humanoid robotics thanks to their strong cognitive skills and learning abilities. Hybrid techniques such as neural fuzzy, fuzzy neural or neural genetics systems and fuzzy genetic systems have large imports in the development of efficient algorithms. These combine the characteristics of the techniques mentioned. Each of them has its own limitations and advantages. Hybrid technology – its integration and synthesis is necessary for the effective use of humanoid robotics because it makes it possible to avoid many of the limitations of the techniques.

4.4.4 Dance in humanoid Robotics

Dance is a largely social activity – whether for esthetic pleasure, entertainment, communication or ceremony – and is likely to be associated socially with our innate. In there are numerous works on robotic dance systems. Researchers at the university of Tokyo have developed the method of learning observation training that allows a robot to gain knowledge of what to do and how to do it by observing human demonstrations. A method called inter modality mapping is applied at Kyoto University to generate sounds from motions using the algorithm for back propagation through time. Tokyo university's other approach is using Chaos in a dancing robot to trade synchronization and autonomy. Built at Tohoku University, Dancing Robot Partner is well Known in the Field of robotics. Their robot acts as a female partner and by estimating their intention, performs ballroom dances in coordination with a male human dancer. While some of the above-mentioned systems are examples where human robot interaction is the key factor in their success, they are exceptions in the robotic dance world. Most robots dance systems are only preprogrammed motions and find them boring after a while for the user. In this work, the proposed system interacts with humans and the motion evolves in accordance with his assessment of the dance section seen.

4.4.5 Evolutionary Computation features

Evolutionary computing is a general computational concept that is biologically inspired and includes genetic algorithms (A), genetic programming and evolutionary strategies. The EC is a search algorithm based on the population and outputs multiple candidates as system outputs, each called an individual. Genetic algorithm is an algorithm of probabilistic search based on natural selection mechanics and natural genetics. Search space is called the space of all feasible solutions. Every search space point represents one feasible solution. Every feasible solution can be marked for the problem by its value or fitness. In some way, the chromosome should contain information about the solution it represents. A binary string is the most common way of encoding. Crossover selects genes and creates a new offspring from parent chromosomes. Mutation occurs after a crossover is performed. This is to avoid falling into a local optimum of solved problem with all solutions in the population. Mutation alters the new offspring randomly.

General GA process is as follows:

Initialize the population of chromosomes

Calculate the fitness for each individual in the population using fitness function

Reproduce individuals to form a new population according in each individual's fitness.

Perform crossover and mutation on the population

Go to step (2) until some condition is satisfied

GA Provides a very effective population based search method and has been applied to many optimization and classification problems.

4.4.6 Interactive Evolution

There are two types of target systems for system optimization: systems that have numerically or at least quantitatively defined optimization performance as evaluation functions and systems that have difficulty specifying optimization indexes. IEC is an optimization method that adopts EC based on subjective human assessment among system optimization. It is simply an EC technique that is replaced by a human user's fitness function. We can say that the IEC is a technology in the target system that incorporates human preference, intuition, emotion, psychological, aspects or a more general term, KANSEI. There are many applications when there is no explicit definition of the fitness function. Interactive evolutionary algorithm uses the witness value of the human response. This allows for the application of algorithms to artistic domains and we propose a dance choreography design aid system for the use of humanoid robots.

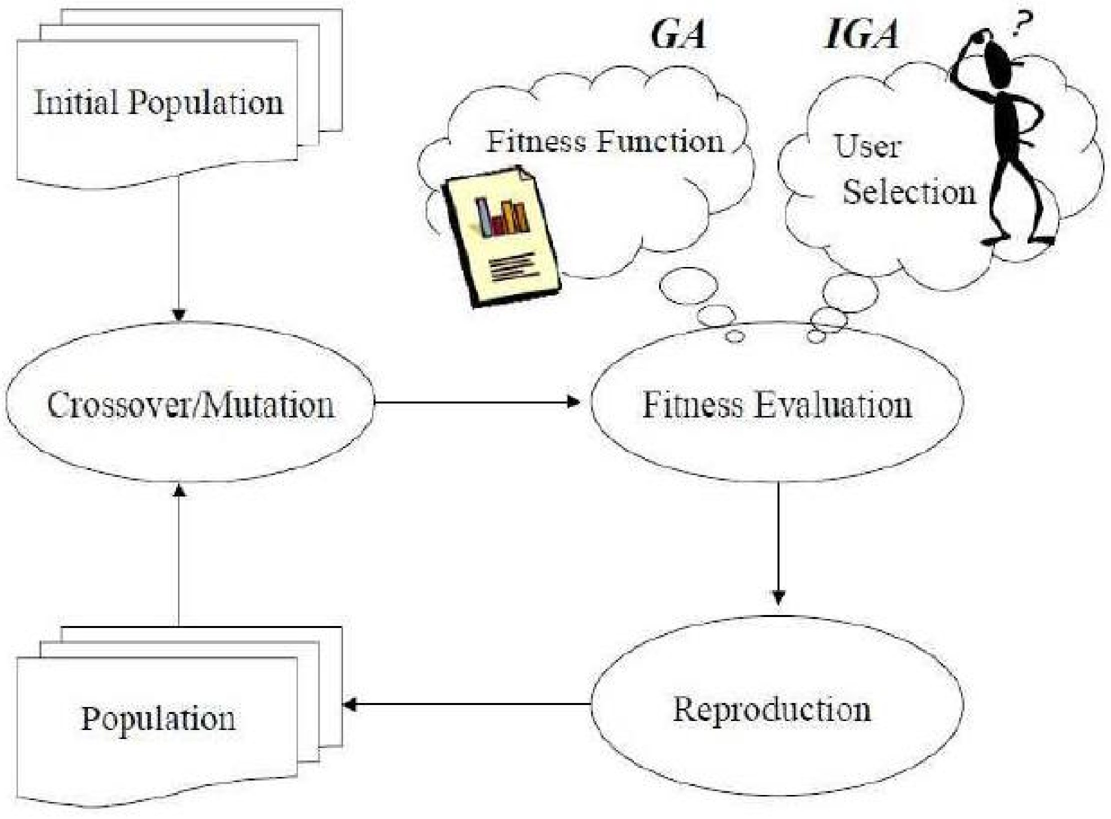


Fig 4.4

4.4.7 Experiment

The experiments are being conducted in Webots, a 3-D simulator of many robots, including the Nao robots. Experimental outputs are dances, so it's very hard to portray them through images. Simulation studies the convergence of dances of each human evaluating them in this set of experiments. In fig 4.5 we can see the third generation of evolutionary postures in dance. Depending on evaluation, the convergence of the algorithm to similar dance motions is evident in 6 – 10 generations. We have not yet included the robot's leg movement. The GUI is user friendly and it is positively estimated by subjects that the proposed system is easy to use. They coincided that system is a useful tool for creating new choreographies of robotic dance.

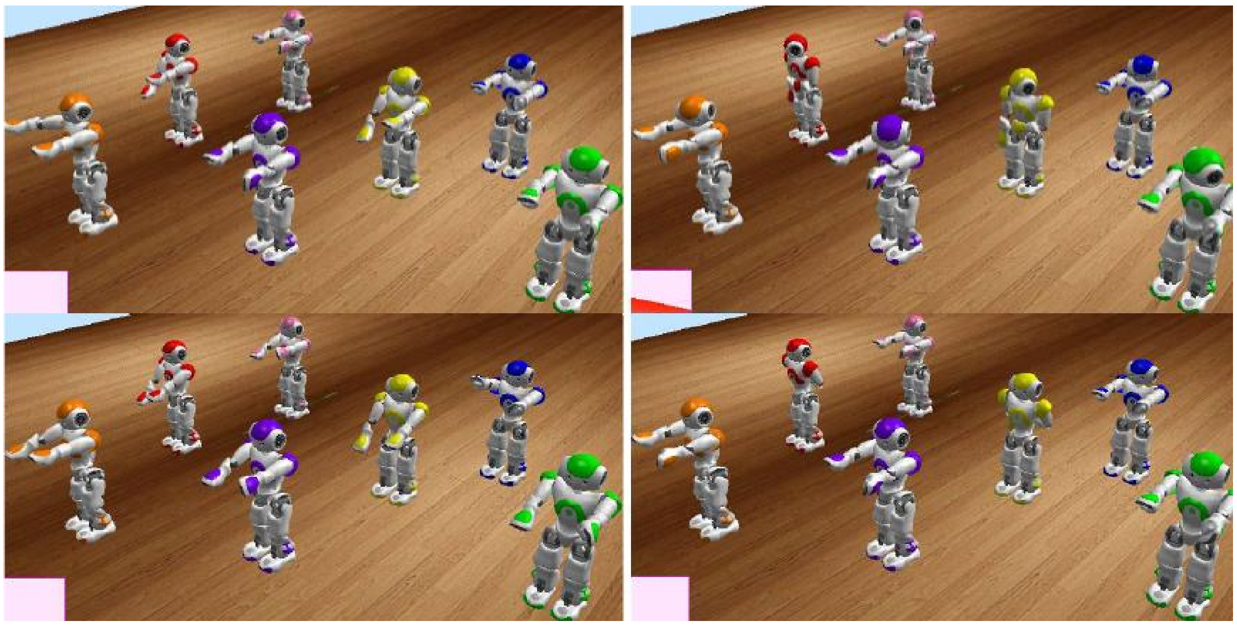


Fig 4.5

4.4.8 Optical Motion of Tracking Server and Robot Simulator Cluster

Human movement analysis has increased its importance and demand as more computationally intensive approaches are available in controlling humanoid robots, creating human motions of CG, and planning medical procedures. In order to investigate the control algorithms of human behaviors and the information processing behind them, it would be useful to measure multi-model data such as foot forces, EMG surface signals, sight directions, pupil diameters and brain wave as well as geometric motion patterns. Real-time applications include, but are not limited to the entire humanoid robot's body motion cockpit.

This paper presents the development of the optical motion capture system developed for the behavior capture studio. The advantage of optical motion capture is the absence of constraints. A subject is only required to wear a suit with many small reflective markers to be observed by multiple camera storage construct 3D motion data. On the other hand, the drawbacks of optical motion capture are:

1. In the common view of the cameras, the motion of the subject is limited and therefore requires considerable environments.
2. Reconstruction of movement data in 3D usually requires human intervention to compensate for mislabeling due to obscured markers.

This paper proposes the pan-tilt tracking camera to solve the first problem. A system of six pan-tilt cameras is developed as a prototype. The computational issue of 3D reconstruction is also discussed in this paper. The 3D reconstruction of the captured data involves identifying the corresponding marker points in multiple camera images, which we call automatic reliable labeling without human assistance.

4.4.9 The Behavior Capture Studio

The behavior Capture Studio Shown in fig 4.6 was constructed. It consists of an optical motion capture system, an eye-mark recorder (NAC EMR-8), floor reaction force sensor plates (KISLER), ultrasonic position trackers (Intersense IS-600), a brain wave/EMG recorder (NEC SYNA ACT MT11), and a 3D image projection system.

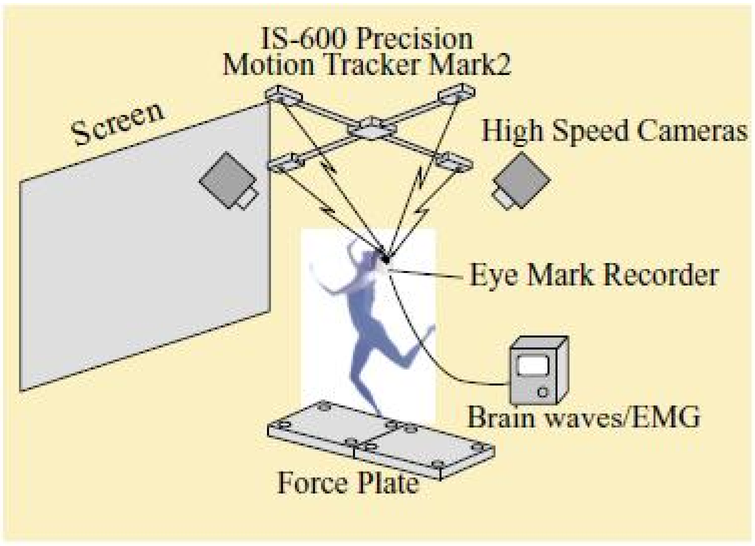


Fig 4.6

4.4.10 Optical Motion Capture

There are two elements in the motion capture system. One is the equipment attached to the subject and the motion information I transmitted, and the other is the equipment set around the subject and the information is received. Passive optical markers correspond to the former in the case of optical motion capture systems, and the latter play a role in multiple cameras. The subject wears a combination of many small retroreflective makers, which reflect light in the direction from which the light came as seen in fig 4.6. a camera unit consists of a 262 fps (512 x 512 pixels, 8 bit) monochrome CCD camera, LED lights and two direct drive motors for pan and tilt axes (fig 4.7) and six camera units are placed on the celling such as fig 4.8. Note that calibration and other geometric and optical parameters determine the detailed camera position and orientation.

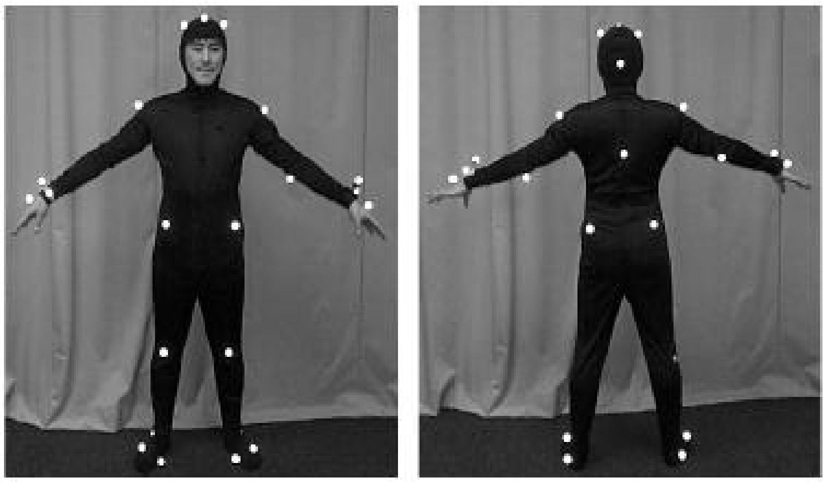


Fig 4.7

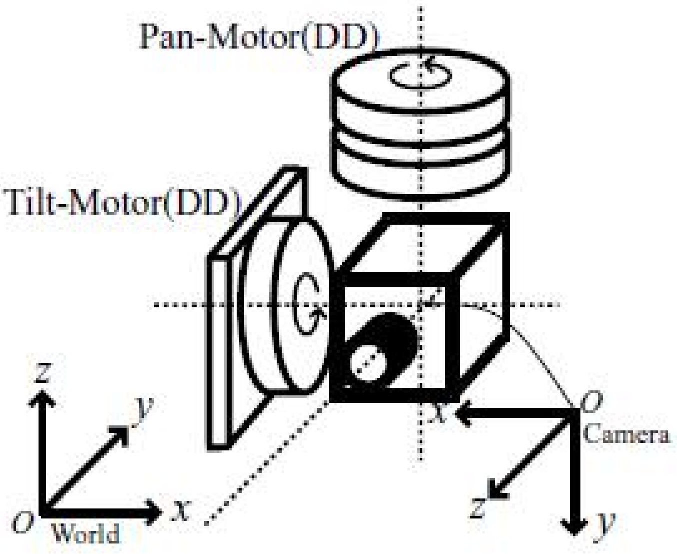


Fig 4.8(a)

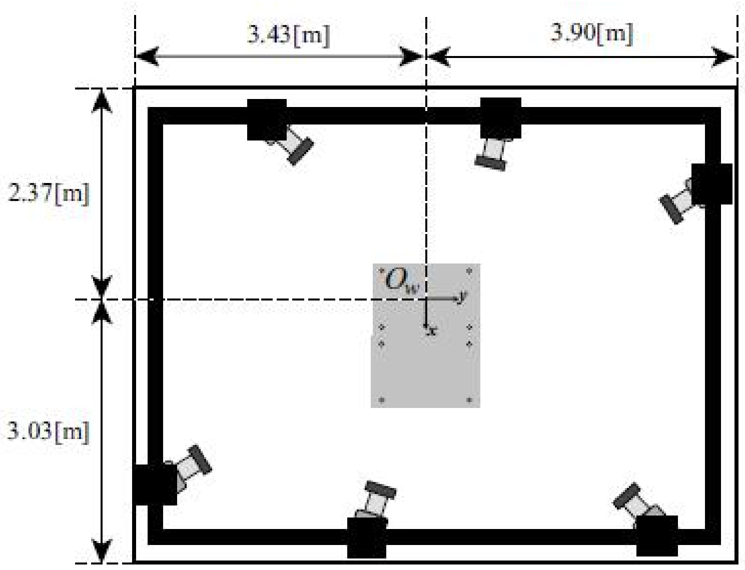


Fig 4.8 (b)

The optical motion capture calculation is based on the following flow:

1. Capturing markers by multiple cameras
2. Marker extraction on the camera image
3. Reconstruction of marker position
4. Marker labeling
5. Computing joint angles

4.4.11 Computer System

Five types of computers are installed and MPI (Message Passing Interface) networked as shown in fig 4.8

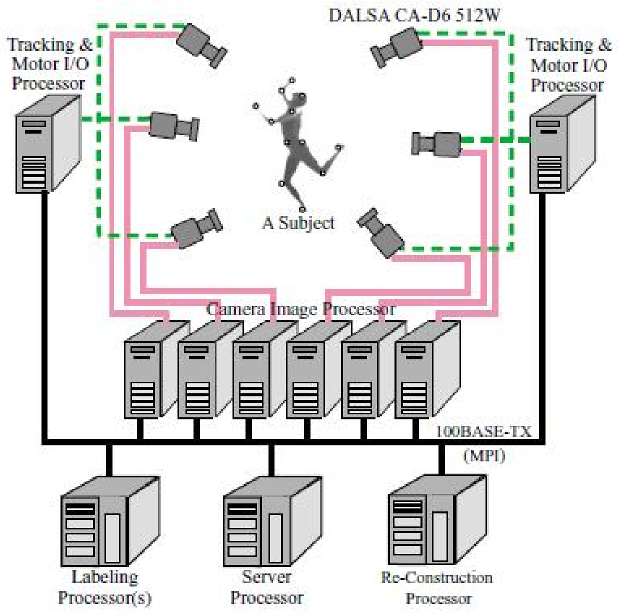


Fig 4.9

Camera Image Processing PC

Each pan-tilt camera is connected to a PC via CORECO VIPER DIGITAL image processing board. This PC processes the image capture sequence, marker extraction and calculation of the marker vector in the local coordinate.

Motor Control PC

Three pan-tilt cameras are controlled by an engine control PC. Each motor driver controls the pan and tilt direct drive motors.

Marker Reconstruction PC

Camera image processing PC marker vector data and camera parameter data from motor control PC are collected and three dimensional marker position is reconstructed.

Server PC

All computers described above are controlled via MPI by this PC.

Labeling PC

All 3D reconstructed markers are identified as well as estimated missing markers.

4.4.12 Real-time Labeling

After obtaining the data on the position of the three dimensional marker, each marker is identified at the request. It is called labeling. Usually, labeling is performed offline, which means that it is performed after capture. In offline labeling, we can optimize the labeling to be consistent at every moment. But such optimization is impossible in the real-time labeling we are proposing. With some appropriate estimation, we should compensate for missing markers and make an effort to do labeling only with the data obtained at each moment. Because we can’t obtain future data in real-time processing, we can’t always rely on past output. We assume that there are some errors in the marker position data. The algorithm is weak against a single error if past output is considered to be completely different. The labeling algorithm should therefore not be designed to inherit potential errors from the past. And the algorithm should be able to determine the position and orientation of each body part in order to obtain fine motion data.

We established Poly-hedra Search Algorithm and A symmetrical Marker Distribution to meet these demands. The first step is to set the unique polyhedral on each body part of the subject. For each rigid body to determine its position and orientation, at least three points are required. Fig 4.10 shows an example marker distribution that each part of the body has at least three markers.

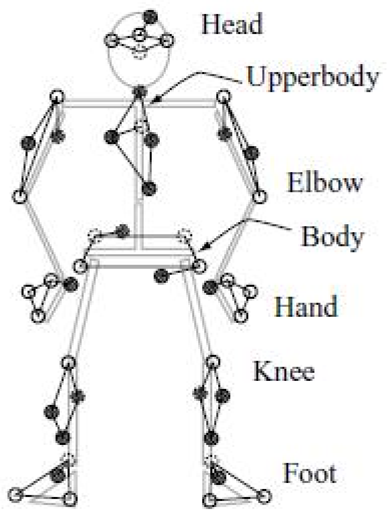


Fig 4.10

The targeted group is searched in the marker position data as a set of markers that are considered to be the vertices of set polyhedron. The procedure shall be as follows:

1. To make a distance table which all the combinations of existing markers are described.
2. Searching for the triangles that make up the targeted polyhedron.

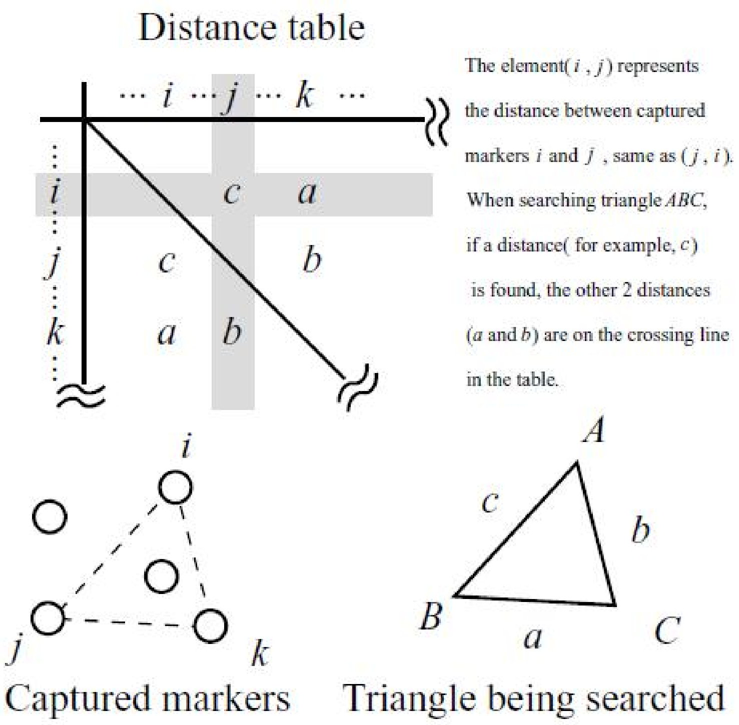
3. Confirm the existence of the other triangle based targeted markers.
4. To check the consistency of the entire body using the model labeling element

Fig 4.11 if some triangles are found, the rest of the markers are searched by the local coordinate spanned by the triangle. For example, when searching for tetrahedron ABCD (fig 4.12), at first the triangle ABC is searched, the local coordinate spanned by p,q, p x q is obtained. The vector to D, denoted by r, is then calculated as follows by the initially known parameters s, t, u:

$$R = s\,\mathbf{p} + t\,\mathbf{q} + n\,\mathbf{p} \times \mathbf{q}$$

S, t, u are calculated at the initial moment when all four positions of ABCD are knowns:

$$\begin{pmatrix} s \\ t \\ u \end{pmatrix} = (\mathbf{p} \quad \mathbf{q} \quad \mathbf{p} \times \mathbf{q})^{-1}\ \mathbf{r}$$


Distance table
… i … j … k …
i
j
k
c a
c b
a b
The element(i , j) represents
the distance between captured
markers i and j , same as (j , i).
When searching triangle ABC,
if a distance(for example, c)
is found, the other 2 distances
(a and b) are on the crossing line
in the table.
i
j
k
Captured markers
A
B
C
a
b
c
Triangle being searched

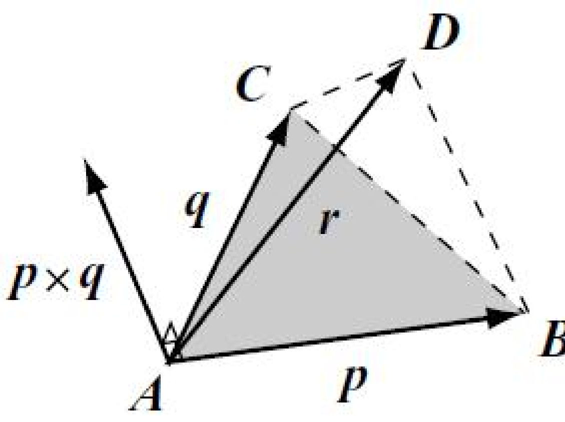
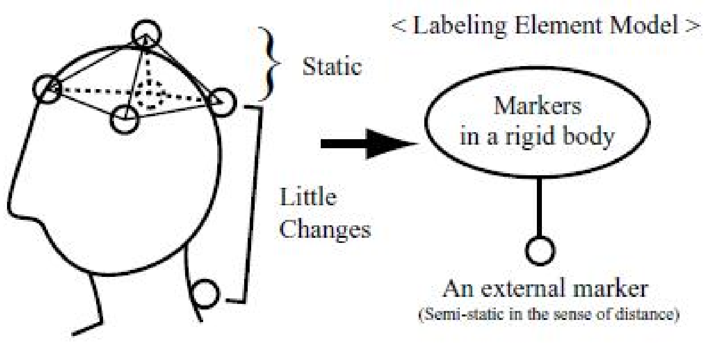


Fig 4.12

Fig 4.11

The same method is available from the fifth marker onwards. Using this algorithm, we can search at least for three markers of each polyhedron to confirm the existence of the remaining markers and estimate the position if certain markers are missing.


< Labeling Element Model >
Static
Markers in a rigid body
Little Changes
An external marker
(Semi-static in the sense of distance)

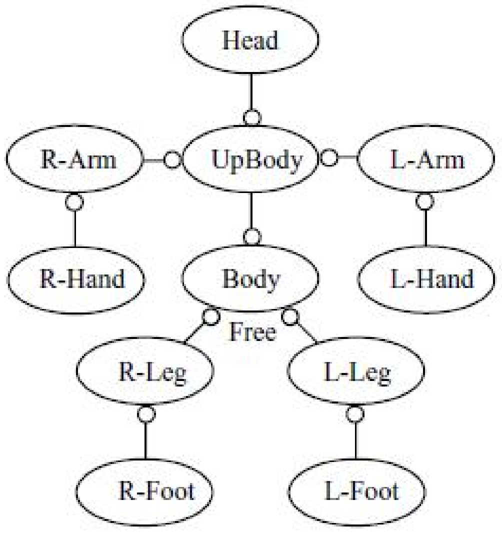


Fig 4.13　　Fig 4.14

There are many combinations in the body parts obtained by the polyhedral search that the spatial relationship is inappropriate as a human body. To put out appropriate labeling result, we should choose the best combination that the relationship between adjacent parts is valid. We now introduce the labeling element model illustrated in fig 4.13. each part is connected together (fig 4.14). the inappropriate connection like head to foot connection is avoided by checking the distance between adjacent parts.

4.4.13 A symmetrical Marker Distribution

If there are many congruent figures in this search method, many candidates are hit in a search and this causes selection of the necessity. It takes a great deal of time. To avoid this, we suggest distribution of symmetric markers as follows:

1. Be careful not to make a symmetrical polyhedral congruent or mirror
2. To be careful not to make polyhedral like triangles of osceles that have ambiguity over the orientation.

We have experimented with labeling some captured subject motion data that have been attached as symmetrically as possible to 20 markers. If all five tetrahedral are congruent equilateral tetrahedral, the number of single polyhedron search candidates is 120. There suit shows a mean number of polyhedron search candidates to be 3.3. we have confirmed that a symmetric distribution of markers can reduce the number of polyhedral search candidates.

When considering the real-time processing of the optical motion capture system. It is important to reduce the computation time for labeling. Since we use DALSA CA-D6 camera capable of capturing at a rate of 262 fps, in the ultimate sense we hope to label it at a rate or faster as well.

5 Chapter 5 MR Related Work and Implementation

5.1 Introduction

In order to achieve an autonomous behavior, many types of functions need to be integrated. There are several developmental environments for robot behavior that include simulation and visualization, for example OpenHRP and Microsoft Robotics Studio. Onishi et al. also developed an immersion – type environment for simulating human robot interaction with virtual reality dynamics. These simulators are used subject to correlation between the simulator and the behavior of a physical robot. The development tools, however, do not include actual environmental factors such as terrain, obstacles, lightning, friction and noise from the sensor. Most sensor inputs are treated by the simulator with noise or ideal noise. Under unknown environments, the factors in the actual environment result in failure of the humanoid behavior. We believe that preparing for the visibility of the internal of the humanoid in the environment is indispensable for the development of the autonomous humanoid behavior in the actual environment.

Mixed Reality MR technology can provide us with a solution to the above problem. Many research studies have studied MR technology and discussed a large number of applications. Using MR technology, the humanoid robot can overlay what it perceives. Plans and controls the real environment around it and its body.

Based on MR technology, footstep planner generated biped foot placements are shown on the captured image in the physical ground surface. Because a motion capture system tracks obstacle positions such as tables or chairs, the footstep planner dynamically calculates adaptive footsteps to avoid obstacles. Developers in motion planning can observe changes in the footsteps on the real environment image captured. In principle, the associated physical position can be projected onto any type of sensed and planned data. This feature is very useful as we can intuitively perceive the data for humanoid debugging.

Furthermore, for humanoid debugging, offline reviewing is important. For instance, developers might want to see the robot’s internal status once the robot behavior is unexpected. The internal data update us more than one kHz, online checking of the data is impossible for them. Offline review function allows us to see the data step by step for such a situation and to find more easily what was wrong.

In this thesis, we introduce an MR debugging system that supports both functions of online viewing and offline review. The system allows both the online status and the recorded status to be viewed in the same view.

5.2 Related work

In the MR field, a few applications have been proposed to operate a robot arm using MR technologies, Klinker, et al., for example, have proposed a system that teaches how to control a LEGO based robot arm with a virtual arm overlay. This application demonstrates the potential of MR technologies in robot operations to improve human friendliness. However, their approach to building as an MR application is too straight forward to apply directly to such a humanoid an integrated complex autonomous robot. Recently, an approach has been proposed using a laser projector to indicate trajectories of industrial robots. In this approach, the laser projector placed on the celling displays the end of the robot hand trajectories onto an actual work piece. This approach makes it easier than conventional ways to understand and confirm the trajectories. There is a great limitation, however, that this system can only render graphics on physical surfaces. In any case, we note that in comparison with a humanoid, an industrial robot is significantly simple, which should move around autonomously.

Humanoid’s Augmented Reality Environment provides online viewing only and our system can provide additional offline review. In addition, their system has the following problem. What type of display is used for MR viewing is very important? A developer must keep watching the humanoid’s behavior carefully to debug a humanoid. However, an ordinal large flat display is used in the system described in to show the results of the footsteps, although we must look away from the humanoid. Our system, on the other hand, allows us to continue to watch the humanoid.

5.3 Approach

The developers want to observe differences between variables in their designed internal models and physical properties of the actual environment when debugging humanoids. The differences cause the humanoid’s unsettled behavior even though the behavior models were well conceived with simulation. The cause of the unexpected behavior in the integrated complex system is difficult to find. The heuristic method related to the actual environment therefore supports developers with Mixed Reality’s graphical representations for efficient debugging. In our approach, the representation of the humanoid can be displayed on the existing functional locations. It provides an intuitive understanding of multidimensional plural information as the inner representation of the behavior of the humanoid. One of the functions is that, from the developer’s arbitrary point of view, sensory measurements of the humanoid are displayed on the actual sensor’s place during the behavior in the actual environment. The 3D graphics, describing the representation of the humanoid, are drawn on the captured image of the viewpoint of the operator, which is related to the pose of a real camera installed in a Head Mount Display (HMD). In addition, we provide the viewing and review system for how the humanoids sensed, planned and behaved. The methods of our approach are described below.

Video See through HMD

For safety reasons, the developer must continue to monitor the humanoid in debugging. Display separation in such a critical operation is not good. Therefore, one of the solutions for presenting the useful debugging environment is the display method for super imposing the information on the representation of the humanoid on the actual scene.

For this case, an HMD is suitable. There are two types of HMD in MR applications. One is a type of optical view through and the other is a type of video view through, the humanoid's graphic representation is many in our approach, and is formed from 3D graphics. The observers can clearly distinguish the overlaid representation on the actual scene using the video see through Head Mount Display VST-HMD

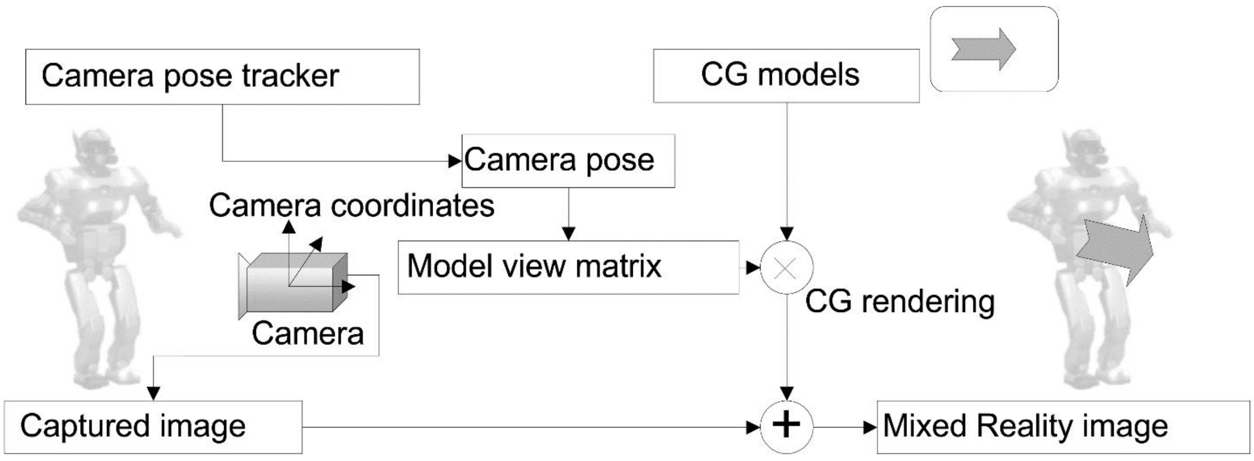


Fig 5.1

Fig 5.1 shows a diagram of deployment to display VST-HMD composite MR image. The VST – HMD is made up of the cameras and display panels for displaying the composite images in front of the operator's eye. To capture the actual scene, the camera is used. The pose of the actual camera just captured the frame is used in rendering 3D graphics as the model view matrix, the pose as a point of view and an eye direction. The camera pose tracking is the topics in current MR research. For this we can use a method of camera pose tracking. Then, the camera pose transforms the models coordinates. Rendering hardware draws as a background the model's 3D graphics on the captured image. Finally, the MR image is displayed in the VST-HMD display panel.

Graphical representation of the humanoid internal

We mentioned the MR visualization method with VST-HMD in the previous approach. The 3D graphics is an important component in debugging, representing the internal process of the humanoid. For example, while an arrow is used to present 3DOF sensory measurements, the measurements can be assigned the arrow’s graphical properties, such as position, orientation and length. Although the graphic’s color and line width can also be used, in transparent view these properties are illegible. In our approach, 6DOF force sensor measurements consists of two arrows, one representing the strength measurement of x, y, and z-axis, and the other representing a few torque along these axes. Other results of internal process, such as footstep planning are prepared for the developer to recognize the variables for 3D graphical representations. The 3D graphical representations are dynamically updated from server notification, which deals with variables of the internal processes of the humanoid.

Furthermore, selecting which contents of viewing’s 3D graphical representations is useful. The developer’s want to change the debugging target’s focus, as there are so many contents/ the scene graph tree implemented in the Open inventor is useful for displaying and controlling the graphical representation. We provide a synchronization management function between the transformation node properties in the tree and the sensory information in the actual behavior. It allows developers in real-time to demonstrate the humanoid’s interactive behavior against their stimulation.

Measure in the same coordinated the actual behavior and the HMD

The position of the odometer of the Humanoid is unstable due to biped walking. It therefore, requires external measurement of the attitude for objective observation. The pose of the HMD is also measured in the same coordinates of the behavior of the humanoid, the device for objective observation. We use an optical motion tracking system to realize this condition. The tracking system can use stereo plural cameras to track certain marker objects. If the transformation between the marker and the humanoid’s physical center is pre-calibrated, the other poses of the humanoid’s body components can be obtained from the joint angles.

Log the behavior of the Humanoid to check

It is practical to reappear the behavior with prepared virtual data when simulating the behavior of the humanoid. Conversely, the behavior under the unexpected environment is difficult to reappear in the actual environments. The developers therefore require the behavior log to be reviewed in the actual environment. The walking control process is too fast to be seen in real time. In some cases, such control behavior is convenient for the developer to review the motion with a slow or reverse replay function. To compare the current behavior, the developer also wants to see the past behavior. For comparison, the past representation should be displayed in the same frame. To meet the demands, we propose a distributed log system. The log system is a simple distributed log module related to each of the humanoid's internal processes. The process output is treated as streams of data. The streams are stored and restored without interfering with the behavior of the current humanoid. The proposed system therefore allows us to view and review the behavior of the humanoid on demand at any time.

5.4 Implementation

VST-HMD Display Device

We use the video view through HMD of a canon type COASTAR, VH-2002 shown in fig 5.2. As one of the key tools for achieving this system. COASTAR stands for See Through Augmented Reality Co – Optical Axis. The VST-HMD has two stereo viewing cameras and two display panels. Each camera and panel pixel resolution is VGA. Since the optical axis of the camera lens center and the panel center are aligned with the straight configuration, we can see a scene naturally even though we need to see through videos. Incase, if we want to handle it remotely then IoT will be introduced for data transmission which differs from the case we considered for our experimental phase. As shown in figure, the motion capture system places several retro reflective markers above the HMD to measure its pose. The local transformation from the origin of the coordinated of the marker to the origin of the coordinated of the camera is pre-calibrated.

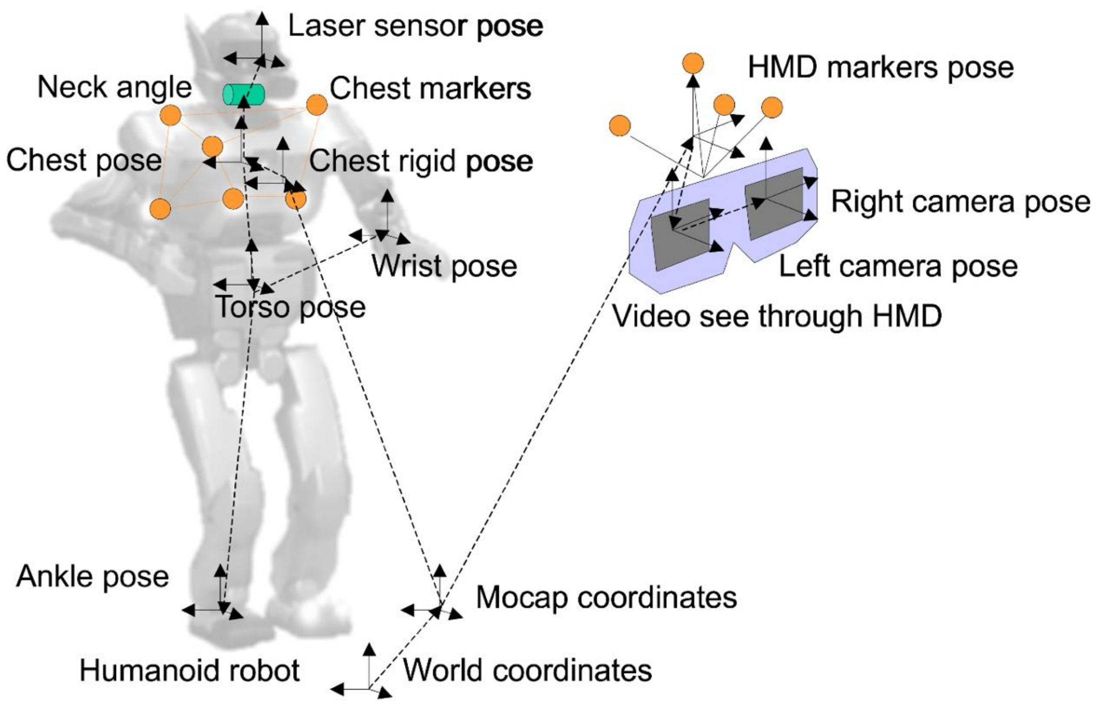


Fig 5.2

Transforms between the HMD and the humanoid

The poses of the humanoid and the HMD camera must be accurately obtained in the actual environment. Between the humanoid and the HMD there are many local transformations as shown in fig 5.2. The HMD markers are described in the previous approach; the humanoid markers are placed on their chest. Once the local transformation between the markers of the chest and the chest body is calibrated, it is possible to describe the attitude constructed by the humanoid's physical joints in the coordinates of the world. In preparing the visualization process of rendering the force sensor measurements, the set of local transformations appearing in the figure is adaptively used to adjust the measurements point of view.

Distributed log system

The processes, such as humanoid perception, planning and control, contain different update frequencies, highly frequent control such as ZMP trajectory generation, less frequent but heavy computation such as stereo vision recognition of the surroundings. By contrast, the display frequency is usually 60 or 30Hz. Managing these processes different frequencies is too complicated to handle like a printout with such a simple log method. The variables reported by the actual humanoid internal processes are therefore treated as data streams shown in fig 5.3 in the proposal system. The streams are established between the processes, distributed log modules, and viewer modules via TCP/IP network protocol over the remote systems. As the internal processes run on the humanoid CPU's, each process sends its own internal variables via FIFO, called pipe, to a respective socket server. Each socket server sends the variables when the updated variables arrive to the connected clients.

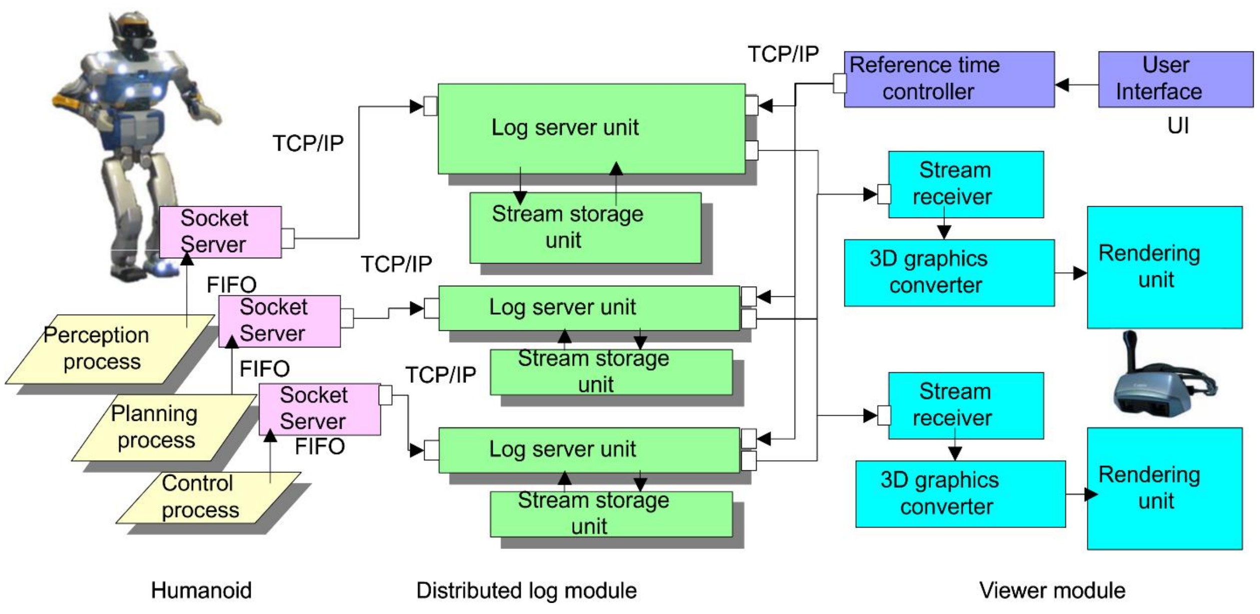


Fig 5.3

The distributed log module is the log system's core function. The log module acts as a proxy server for network caching. It consists of log server and a storage unit for stream. There are three types of socket port on the server unit. One port is connected to the humanoid's socket server to get the internal process variables. If possible, the log unit will bring it to the storage unit when the socket server sends the updated variables. The storage unit of the stream has a dual method of access. The storage unit can pick up the reference past data while adding new data without interference. The data is serialized by the storage unit to files during behavior interruption. The log server unit's control socket port waits from a reference time controller for a reference time pocket. The reference time packet refers to the data timestamp, which is related to the viewing of an observer. If the viewer module connection is established, the log unit collects the data requested by the reference time controller from the storage unit. Then the log unit immediately forward the data to the viewer's module. A user interface, like a jog dial, operates the reference time controller. It allows slow or reverse checking by checking the reference time packet sequence.

Plural log modules can be increased on the request and the modules can be controlled by multiple reference time controllers. The system is expandable and flexible as all modules accept multiple client connections.

Viewer Module

The module of viewer is a stream receiver, a 3D graphics converter, and a rendering unit. Under the 3D graphics converter, the stream receiver works to hold current values sent by either the log module or the socket server. The converter unit prepares 3D graphical data before rendering when the rendering unit is updated to redraw the current mixed reality image. The converter unit also calculates the current poses with the measurements of motion capture to adjust the attitude and the status of the sensory measurements. The rendering unit contains a scene graph tree of the models of the humanoid. Visual effects of shadow rendering and texture mapping of the environment are used in the rendering unit for realistic visualization. It is also possible to connect the viewer module to the humanoid’s plural log module and socket server. Comparing visualization with plural server connections, the system can display the graphical representations in real time and/or in the past as the plural humanoid’s behavioral internals.

6 Chapter 6 Challenges

As we need connectivity between two servers and advanced humanoid to do our required task so challenges will be there during implementation of this system.

6.1 Humanoid Complexity

Humanoid Robot is one of Artificial Intelligence more specifically deep learning application and Robotics in general's newest emerging and challenging field. If a simple robotics with kinematics is complex then the humanoid we suppose in our thesis is way more complex because we not only need that robot but we also want it to be connected to MR as well as also need to have a database to store the events it is doing for other mode in which it needs to think by itself and make decisions. Over the past few years, this field has received significance so far so improvements have been seen but couldn't make fully autonomous system. That's why, our customized robot still needs to be manufactured. As defined by Hirai et al., 1998 and Hirukawa et al., 2004 Hominoid robots are human body – based robots with an overall appearance. Humanoid, in other words, refers to any being whose body structure resembles that of a human being; head, torso, legs, arms and hands. Yet it is also made in appearance, actions and behavior to resemble a human. In recent popular movies like AI Artificial Intelligence (2001) by Steven Spielberg, Ex Machina (2015) by Alex Garland, The Machine (2013) by Caradog W, Humanoids robots have been seen recently. In our case we want a humanoid complex enough that can be controlled remotely using Mixed Reality (MR). This complex robot will be able to switch between two modes. One is imitating and learning and the second one is, based on database inside it, will think and decide to function the tasks required. It will be able to make maps to accomplish required goals in factory. On the other hand, a huge change on humanoid robots has been witnessed in recent years. They are designed to think and act just as human beings do. They are designed with a special neural system scheme so that they can operate independently without any human intervention. They also have a sense of perception that enables them to adapt to both the human environment and the environment of machines. Because their CPU is quite advanced in its form, they are made to act very fast and therefore give the maximum throughout that previous robots have never had before. One good thing about them is their ability to grasp; they are made to learn things so easily, so in their learning faculty, not much programming is needed as they can automatically learn on their own. These robots are used as receptionists in homes, offices, companies or as customer representatives in telephone companies.

Millions of inputs and outputs are received and sent. Some robots are said to be even connected to knowledge – based system known as the Blue Book or we can say database, something more like a search engine that processes all the answers whenever questions are put to the machine. It is very important to note that if programming is good then there is a high likelihood of the machine performing beyond expectations. The algorithm's complexity plays an important role in this aspect in most cases. In addition, there is the actuator in a mechanism for activating process control devices using pneumatic, hydraulic or electronic signals. The actuators are thus used to supply strength and energy to all joints connected to each other. Another key component is the wireless receiver that allows direct access to the machine or allows the machine to be connected to both local and global networks. Some of their examples include the popular ASIMO from Honda, first launched in October 2000. Moreso, another massive robot that history will never forget is SARCOS; this robot has actually been designed in several U.S universities and research Institutes for research and entertainment purposes.

Humanoid robots offer new questions about how to use them in an effective way of learning compared to simple, wheel – equipped vehicles. One of the most relevant challenges is related to the relatively low level of abstraction provided by their programming environments. This is mainly due to the complexity of the robotic structure: the relatively large number of degrees of freedom does not allow the level of programming that can usually be found in simpler types of robots such as robot cars to easily accessible. We will show this using the humanoid Robovie-X and then propose a higher level of abstraction through a programming interface for natural language.

6.2 Controllability

The control we need has been discussed in diagrams before. In one of the previous projects, the challenges faced by DRC teams attempting to use the Oculus Rift DK1 was that no input device was built. While viewing the sensor data in 3D was considered potentially useful, the problem was that the other interface functions, such as the robot command, were still on the 2D interface. To be effective, the VR interface must have both the robot's visualization and control within VR. We also wanted to translate the movement of the operator's arms, hands and fingers to those on the robot, so that the correlation between the operator's movement and the intended robot movement would be more apparent and direct. This translation involves too much robot automation, as the operator does not only control individual joints but also defines higher level trajectories that will help robot in its movement and do its functions.

6.3 Connectivity with humanoid

Many manufacturing plants around the world uses industrial robots but never considered humanoid with Mixed Reality. Those product class in industry has reached a high maturity level and there is a wide variety of robots available from different manufacturers for special applications. While both industrial and humanoid robots manipulate objects and the same component types, e.g. harmonic drive gears, can be found in both types, the target systems differ considerably. In secluded environments, industrial robots operate strictly separated from humans. They perform a limited number of repetitive tasks that are clearly defined. These machines are often designed for special purposes and the tools they use. Typical development goals are high accuracy, high payload, high speeds and rigidity. Humanoid robots work together with people in a shared space. They are designed as universal assistants and should be able to learn new skills and apply them to new tasks that were previously unknown. Humanlike cinematics enables the robot to act in an environment that was originally designed for humans and similarly use the same tools as humans. Human appearance, behaviors and motions familiar to the user through peer interaction make humanoid robots more predictable and increase their acceptance. Safety is a critical requirement for the user. A lightweight design is not only important for the mobility of the system, but also for the safety of the user as a heavy robot arm I likely to cause more harm in the event of an accident than a lightweight and more complaint one. Much of the development knowledge and product knowledge of industrial robots cannot be applied to humanoid robots due to these significance differences. It is not possible to fully simulate the multi modal interaction between a humanoid robot and its environment, human users and eventually other humanoids in its entire complexity. Real humanoid robots and experiments are needed to investigate these coherences. Only toy robots and a few research platforms are currently available commercially, often at high cost. Most humanoid robots are designed and built according to a particular research project’s specific focus or goals and many more will be built before mature and standardized robots at lower prices are available in larger numbers. Knowledge gained from the development of industrial robots that have been used for decades in industrial production applications cannot simply be reused in the design of humanoid robots due to significant differences for both product classes in target systems. Companies have developed a few humanoid robots, but not much is known about their design process and there is rarely available information that can be used to increase time and cost efficiency in the development of new improved humanoid robots. A humanoid robot design is a long and iterative process as there are different interactions between, for example, mechanical parts and the control system. The aim of this article is to help shorten the development time and reduce the number of iterations by presenting an efficient design process, a method for optimizing light yet rigid support structures and presenting the design of humanoid robot ARMAR III's upper body

6.4 VR Gloves & VR glasses Interfacing

Either the Manus VR glove or the HTC Vive controllers are the two main types of controllers that the operator can use to interact with the virtual objects to manipulate them, such as camera windows. With more buttons, the default controllers allow for more options to enable and disable specific features. The Manus glove's primary advantage is the ability to control the fingers directly. With the finger tracking enabled, the robot will imitate as shown I fig 5.1 as the operator opens or closes the irrespective fingers. This allows for a much finer grasp than a simple open and close configuration, while keeping the usage speed much faster than the keyboard manually controlling each finger. There are two ways the operator can send commands to the robot to control the arms. The first is by pressing a trigger button, then the robot moves its respective hand to the location of that controller. You can hold down the trigger button while moving your hands, which causes the robot to continuously match your movements with its own. While it is useful for quick movements where forecasting is not a priority. It relies entirely on the reverse cinematics solution for movers.

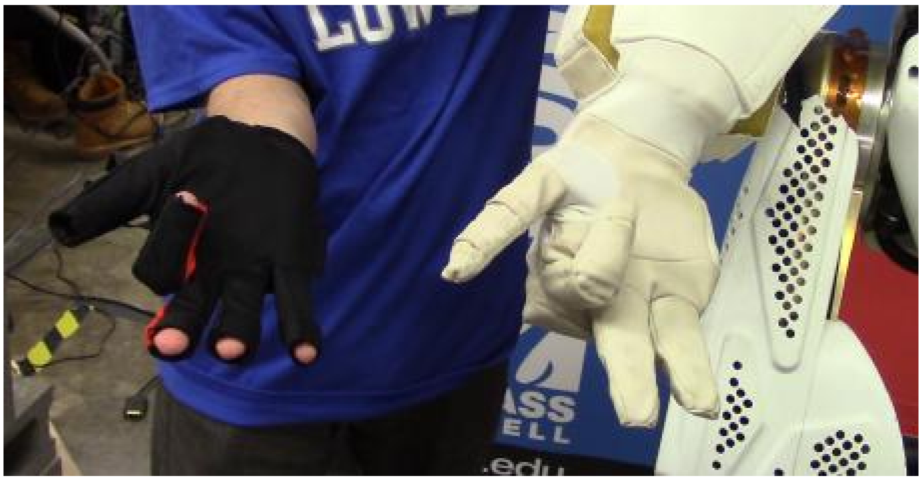


Fig 5.1

The second way is to grasp the virtual limb on a joint basis and move it into the configuration that the operator wants. The real robot will then move to the correct configuration after confirmation. These two methods provide a balance between speed and accuracy and were both methods used in DRC. A fairly straight forward joystick command is the current method for commanding the robot will continue to move forward, with the length of footprint correlating to the joystick's magnitude. The 3D environment while being a simple design. Alternatively, by pointing to a location in the world and pressing a trigger, the operator can command the robot to walk, then the robot will plan the steps to the location. A third solution proposed is to enable the operator to place virtual objects as planned footprints, and then the robot would try to take those exact steps. As with arms manipulation, the combination of these three methods should enable the operator to choose between speed and precision based on the task needs. The primary method for controlling the head currently involves the operator grabbing the head, similar to grabbing interactive windows, and pulling away where the operator wants the robot to look, an example of which can be seen in fig 5.1 an alternative strategy, whereby the operator could control the head in a direct control method, was considered and implemented. The head would look in whatever direction the operator looked in the virtual world by turning on an egocentric mode.

7 Chapter 7 Simulations

As the simulation is needed to get the results as the system is of factory level so developing whole system for demonstration needs time. We have broken the simulation of thesis in few parts. Instead of using a system of cameras for capturing agent actions so the robot can mimic such actions in factory we used different device so we can manipulate the data taken from a device. The device we used for simulation is KINECT which is best choice as far as the prototype of our system is concerned. We have added this device to increase the option of mimicking agents. In case, we don't want to imitate HMD wearing agent we can simply consider another agent and it gets easy to deal with using KINECT.

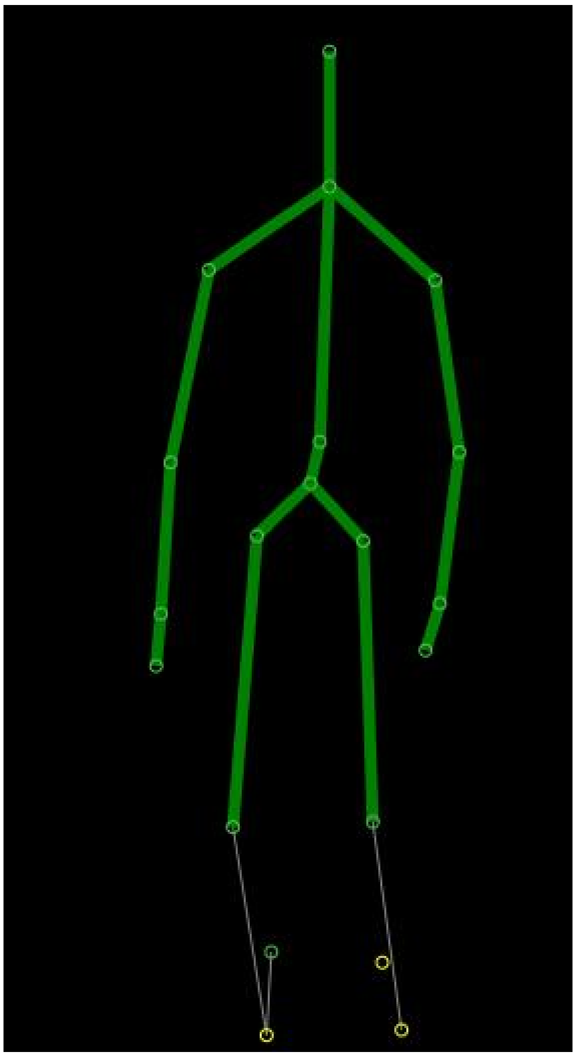
$$\begin{tikzcd}
 & \text{Data imitation} \\
\text{Data acquisition} \arrow[ur] \arrow[dr] & \\
 & \text{Database usage}
\end{tikzcd}$$

7.1 Data Acquisition

As we know we need to get the data to manipulate where points of an agent are located which a robot needs to imitate. When an agent is moving and doing some actions, for example picking up some packages in a factory and placing on some table, we will be using KINECT device to get that data for us. So, there will be 20 points to manipulate. These twenty points will be as follows showing different parts of the body of an agent;

- *Head*
- *Neck*
- *Shoulder left*
- *Elbow left*
- *Wrist left*
- *Hand left*
- *Shoulder right*
- *Elbow right*
- *Wrist right*
- *Hand right*
- *Spine*
- *Hip center*
- *Hip left*
- *Knee left*
- *Ankle left*
- *Foot left*

- *Hip right*
- *Knee right*
- *Ankle right*



- *Foot right*

The above points shown in the image is taken in simulation using visual studio and twenty points have been identified showing unique body parts as we have listed before. A program is c plus plus is written to fetch the data. We connected the KINECT at the beginning and checking how many KINECT devices have been attached. Once we know through the code how many devices for motion detection has been identified we initialize the device in a program. As we know that we need to fetch twenty points of the body and make skeleton through the data so we will specify through the code that we will be using skeleton. After that we need to handle the data by smoothing it. We will ensure the Direct2D to draw the skeleton as we drew in above picture. We will then start drawing the skeleton as we wanted. In case, we lose the device or detecting data we will recreate the render target by disposing it and retry drawing. We will render the torso by creating head, neck, spine, hip, left shoulder, left elbow, left wrist, left hand, right shoulder, right elbow, right elbow, right wrist, right hip, right knee, right ankle, right foot, left hip, left knee, left ankle, left foot. We will draw these points to show these body parts.

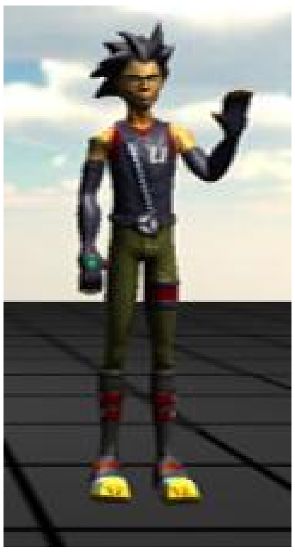
The approach to draw the left arm will include points for left shoulder, left elbow, left wrist and left hand. To draw right arm will include points showing right shoulder, right elbow, right wrist and right hand. Left leg will use the points that will be showing left hip, left knee, left ankle and left foot. Right leg will include points that will be showing right hip, right knee, right ankle and right foot. After that we need to draw the joints in different color and make bone line between two joints. Followed by all we will convert the skeleton to screen space.

7.2 Data imitation

According to what we need in our simulation was to use the data. As we have discussed before we fetched the data showing different body parts. Those parts have specific spatial locality specifying required characteristics for our customized robot so it can use it to function first mode which was to imitate the user. As we know we need a robot which can use this data and experiences forward kinematics and reverse kinematics continuously. In our simulations, instead of using a customized robot, which can use the database as well for artificial intelligence, we used UNITY3D for simulation. We used a virtual robot for user imitation. Whatever the robot we like to import we can be imported to accomplish the task. Whatever task the user was doing a virtual robot follows it. In this way, we accomplish the first mode.

7.3 Database usage

As our robot needs to have a hybrid behavior means it needs to follow the user in case when user was doing some tasks such as picking up some package etc. as well as to use the database. This database will be responsible for storing the data which robot needs to follow in second mode. Now the concept of deep learning will be induced in which the robot needs to think on its own. All robot needs to have is the database specifying how the tasks has been done by the user before so it has to imitate but independently. This mode needs artificial intelligence in a sense that it becomes capable to encounters the hurdles placed in the path of its planned task. For example, if robot needs to pick the object from point A and place it on to point B so it will dedicate some track. In case, it encounters some objects in its path then it will change its planned pathway to second best pathway. In order to do so, it will need a lot of sensors to detect such barriers and need a lot of artificial intelligence to map the pathway to its destination. As a result, we can say experience based learning will be done by our robot. In the following, we used KINECT to show movements of user captured by it. Second mode with database involved in it can't be used by UNITY plateform.



8 Chapter 8 Conclusions

We presented a development system for a humanoid with autonomous system in this thesis. The mixed reality system allows us to view and review how the humanoid’s internal processes sensed, perceived, planned and behaved. The system features, online viewing and offline review are useful for intuitively understanding how each module in the humanoid affects the overall system and how one is affected by the surrounding environment of the humanoid.

With a practical development, the system’s effectiveness is not demonstrated quantitatively. Because it is not possible to collect enough samples of developmental cases, the overall system is too expensive to prove the efficacy with general cases. We are currently expanding the system as a part of a development platform due to the use if a general robot. The technology implemented in the system appears to be useful for general human robot interaction HRI in addition to development. Everybody can see what and how robots think by wearing an HMD. The feature as communication equipment can be an alternative to other approaches such as controlling humanoid facial expression. Only visual information was used by the current system, with other sensory channels it is possible to improve the perception of the developer. Tangible interface is a good candidate for bi-directional communication for the system is for humanoid development. Through practical humanoid development we will examine the effectiveness.

It is possible to improve and extend the work presented in this thesis in many ways. For future work, we provide an indication of the most promising directions.

Our work in controlling the robot focused exclusively on control of kinematics based on the data we provide to it. It also depends on the customized robot as well which will be best suited for hybrid functionality. If a detailed physical model of the full robot is constructed, it will be next step in the field of artificial intelligence to make more revolutions using this robot. Interestingly, an algorithm could learn such a model from experience. The algorithm needs to be accelerated even more in order to apply NOC control to such dynamic control. This appears to be a promising possibility with the introduction of faster simulations, parallel processing and GPU acceleration.

To effectively measure the position of the contact points and the corresponding contact force, an artificial skin can be added to the HRP-2 sole on a mid-term basis. Then the simulation approach developed in this thesis can be applied directly to the actual robot. A second perspective is integrating the KINECT into the sensing of the environment. Alternatively, the 3D environment can be scanned by a LIDAR as it does not depend on lighting conditions as the robot stereo pair does, and can therefore, be used for a wide range of environments. To have a more robust and meaningful reconstruction of the environment, the measurements can be fused with the information obtained from the stereo pair.

9 Literature Review

[1] F.Kanehiro. K. Fujiwara, S. Kajita , K.. Yokoi, K. Kaneko, H. Hirukawa. Open 8] Architecture humanoid robot platform in Proc. of ICRA 2002, pp. 24-30, 2002

[2] Microsoft Robotics Studio – http://msdn.microsoft.com/robotics/

[3] M. Onishi. T. Odashima, F. Asano, and Z. LUO. Development of PC Based 3D dynamic human interactive root simulator. In Proc. of CIRA 2003, pp 1213-1218, 2003

[4] M. Onishi , T. Odashima, Z. Luo and S. Hosoe. 3D immersion type dynamic simulation environment for develpoping human interactive robot. In video Proc. of ICRA 2004, 2004

[5] j. Chestnutt, P. Michel, K. Nishiwaki, M.stilman, S. Kagami, and J KKuffner. Using real time motion capture for humanoid planning an algorithm visualization. In Video Proc. of ICRA 2006, 2006

[6] K. Kaneko, F. Kanehiro, S.Kajita, H. Hirukawa. T. Kawasaki. M. Hirata, K. Akachi and R. Isozumi Humanoid robot HRP-2. In Prec. Of ICRA 2004, pp, 1083-1090, 2004

[7] A Takagi, S. yamazaki, Y. saito, and N. Taniguchi Development of a stereo video see through HMD for AR systems. In Proc. of ISAR 2000. Pp. 68-80, 2004

[8] J. Chestnutt and J. Kuffner. A tiered planning strategy for biped navigation. In Proc. of Humanoids2004, 2004

[9] K. W. Arthur, K. S. Booth, and C. Ware. Evaluating 3d task perfor- mance for fish tank virtual worlds. ACM Transactions on Information Systems, 11(3):239–265, july 1993.

[10] L. G. Carlton. Control processes in the production of discrete aiming responses. Journal of Human Movement Studies, 5:115–124, 1988. [3] R. Chua and D. Elliott. Visual regulation of manual aiming. Human

[11] Movement Science, 12:365–401, 1993. [4] D. E. Meyer, J. E. K. Smith, S. Kornblum, R. A. Abrams, and C. E.

[12] Wright. Optimality in human motor performance: ideal control of rapid aimed movements. Psychological Review, 95:340–370, 1988.

[13] A. Murata and H. Iwase. Extending fitts’ law to a three-dimensional pointing task. Human Movement Science, 20(6):791–805, Dec. 2001.

[14] A. Steed. A simple method for estimating the latency of interactive, real-time graphics simulations. In Proc. ACM VRST, 2008 [In Press].

[15] C. Ware and R. Balakrishnan. Reaching for objects in vr displays:lag and frame rate. ACM Transactions on Computer-Human Interaction, 1(4):331–356, Dec. 1994. [8] R. S. Woodworth. The accuracy of voluntary movement. Psychological

[16] R. S. Andersen, O. Madsen, T. B. Moeslund, and H. B. Amor. 2016. Projecting robot intentions into human environments. In IEEE RO-MAN.

[17] RaviTejaChadalavada,HenrikAndreasson,RobertKrug,andAchimJLilienthal. 2015. That's on my mind! Robot to Human Intention Communication through on-board Projection on Shared Floor Space. In ECMR.

[18] T. Chakraborti, S. Kambhampati, M. Scheutz, and Y. Zhang. 2017. AI Challenges in Human-Robot Cognitive Teaming. CoRR abs/1707.04775 (2017).

[19] TathagataChakraborti,SarathSreedharan,AnaghaKulkarni,andSubbaraoKamb-hampati. 2017. Alternative Modes of Interaction in Proximal Human-in-the-Loop Operation of Robots. CoRR abs/1703.08930 (2017).

[20] Henrik I Christensen, T Batzinger, K Bekris, K Bohringer, J Bordogna, G Bradski, O Brock, J Burnstein, T Fuhlbrigge, R Eastman, et al. 2009. A roadmap for us robotics: from internet to robotics. CCC and CRA (2009).

[21] CatherineDiaz,MichaelWalker,DanielleAlbersSzafir,andDanielSzafir.2017. Designing for Depth Perceptions in Augmented Reality. In (To Appear) ISMAR.

[22] Gal A Kaminka. 2013. Curing robot autism: a challenge. In AAMAS.

[23] Florian Leutert, Christian Herrmann, and Klaus Schilling. 2013. A spatial augmented reality system for intuitive display of robotic data. In HRI.

[24] IñakiMaurtua,NicolaPedrocchi,AndreaOrlandini,JosedeGeaFernández,Chris- tian Vogel, Aaron Geenen, Kaspar Althoefer, and Ali Shafti. 2016. FourByThree: Imagine humans and robots working hand in hand. In ETFA. IEEE, 1–8.

[25] Shayegan Omidshafiei, Ali-Akbar Agha-Mohammadi, Yu Fan Chen, N Kemal Ure, Jonathan P How, John Vian, and Rajeev Surati. 2015. MAR-CPS: Measurable Augmented Reality for Prototyping Cyber-Physical Systems. In AIAA ARC.

[26] ShayeganOmidshafiei,Ali-AkbarAgha-Mohammadi,YuFanChen,NazimKemal Ure, Shih-Yuan Liu, Brett T Lopez, Rajeev Surati, Jonathan P How, and John Vian. 2016. Measurable Augmented Reality for Prototyping Cyberphysical Systems: A Robotics Platform to Aid the Hardware Prototyping and Performance Testing of Algorithms. IEEE Control Systems 36, 6 (2016), 65–87.

[27] Shin Sato and Shigeyuki Sakane. 2000. A human-robot interface using an interactive hand pointer that projects a mark in the real work space. In ICRA.

[28] SeanO'Kane.2015.MicrosoftusedthisadorablerobottoshowoffnewHoloLens features. https://goo.gl/bBSqn3. (2015). The Verge.

[29] Jinglin Shen, Jingfu Jin, and Nicholas Gans. 2013. A Multi-view camera-projector system for object detection and robot-human feedback. In ICRA.

[30] Daniel Szafir, Bilge Mutlu, and Terrence Fong. 2014. Communication of Intent in Assistive Free Flyers. In Proceedings of HRI. ACM, 358–365.

[31] Daniel Szafir, Bilge Mutlu, and Terrence Fong. 2015. Communicating Directionality in Flying Robots. In Proceedings of HRI. ACM, 19–26.

[32] MatthewTurkandVictorFragoso.2015.ComputerVisionforMobileAugmented Reality. In Mobile Cloud Visual Media Computing. Springer, 3–42.

[33] AtsushiWatanabe,TetsushiIkeda,YoichiMorales,KazuhikoShinozawa,Takahiro Miyashita, and Norihiro Hagita. 2015. Communicating robotic navigational intentions. In IROS. IEEE, 5763–5769.

[34] Tom Williams, Gordon Briggs, Brad Oosterveld, and Matthias Scheutz. 2015. Going Beyond Literal Command-Based Instructions: Extending Robotic Natural Language Interaction Capabilities. In Proceedings of AAAI.

[35] Yu Zhang, Sarath Sreedharan, Anagha Kulkarni, Tathagata Chakraborti, Hankz Hankui Zhuo, and Subbarao Kambhampati. 2017. Plan Explicability and Predictability for Robot Task Planning. In ICRA. IEEE.

10 Appendix A

HMD	Head Mounted Display
MR	Mixed Reality
COMAN	COMplaint HuMANoid Platform
ATR	International Advanced Telecommunications Research Institute)
NREC	National Robotics Engineering Center
CHIMP	CMU Highly Intelligent Mobile Platform
COMAN	COMplaint HuMANoid Platform
NREC	The National Robotics Engineering Center
DRC	DARPA Robotics Challenge
VRC	Virtual Robotics Challenge
HRI	human robot interaction
MPI	Message Passing Interface
NOC	Network operating center

10.1 Chapter 1

First humanoid robot

World famous ASIMO

Humanoid Robotics Project HRP

10.2 Chapter 2

Virtual reality

Evolution towards the new era

Mobile First to AI First

Kinematics

Traditional approach to Inverse Kinematics

Jacobian based approaches

Parallelizing the algorithm

10.3 Chapter 3

Framework for movement generation

Interface layer

Control layer

Service layer

Infrastructure layer

10.4 Chapter 4

Environmental boundaries

Robotic grasping

Use of Artificial Intelligence Methods in Humanoid Systems

Optical Motion of Tracking Server and Robot Simulator Cluster

Interactive Evolution

Capturing markers by multiple cameras

Marker extraction on the camera image

Computing joint angles

Humanoid robot interaction

Real time labeling

Camera Image Processing PC

Motor Control PC

Marker Reconstruction PC

Server PC

Labeling PC

Asymetrical Marker Distribution

10.5 Chapter 5

Autonomous behavior

Several developmental environments

Simulating human robot interaction with virtual reality

Footstep planner generated

10.6 Chapter 6

Humanoid Complexity

Human environment and the environment of machines

Controlability

Challenges faced by DRC

Robot visualization and control within VR

Connectivity with humanoid

Humanlike cinematics

10.7 Chapter 7

KINECT

Direct2D

Body parts

10.8 Chapter 8

MR allows us to view and review

Humanoid internal processes sensed

Hybrid functionality

Scanned by LIDAR

11 独创性声明

12 学位论文使用授权说明