OPTIMIZING UPPER BODY MOTION OF HUMANOID ROBOT USING HIERARCHICAL B-SPLINE

階層的 B スプラインを用いたヒューマノイドロボットの上半身動作の最適化

by

Miti Ruchanurucks

A Master Thesis
修士論文

Submitted to
the Graduate School of Information Science and Technology
the University of Tokyo
on January 28, 2005
in Partial Fulfillment of the Requirements
for the Degree of Master of Information Science and Technology
in Information and Communication Engineering

Thesis Supervisor: Katsushi Ikeuchi 池内　克史
Professor of Graduate School of Interdisciplinary Information Studies

ABSTRACT

Recently there are various kinds of research on humanoid robot, such as motion control, man-machine interface, visual or haptic feedback, artificial intelligent, and so on. For motion control, the present works still have many problems since the number of degrees of freedom (DOFs), range of joint motion, achievable joint velocity or acceleration, and dynamic motion capability of today's humanoid robots are far more limited than those of the average human subject.

Our research group focuses on motion control especially using robot to preserve Japanese traditional dances, which are considered as intangible cultural assets. In this work the upper body motions of robot is generated. First the original human motion is acquired by motion capture system. Then inverse kinematics is used to convert data in the form of markers' positions into joint angle of robot. As the robot has many physical constraints, converted data often cannot be used directly, so the optimization process is used. The optimization algorithm's objective function is subjected to preserve salient characteristic of the original motion, while constraints for optimization are used to transform the motion of human to the capabilities of the humanoid robot. Hierarchical B-spline is used as data representation for trajectory optimization. The B-spline function is exploited in all constraints for optimization, which makes our work different from previous trajectory optimization based methods that only reduce robot's physical parameters such as velocity or force, but not effectively limit them.

Generated motions are first simulated in the program developed by our research group, then the humanoid robot HRP2 is used to test optimized motions.

Acknowledgements

First, I would like to express my sincere gratitude to my advisor, Prof. Katsushi Ikeuchi, for his kindness that would be invaluable to my future.

I am very grateful to Dr. Atsushi Nakazawa Dr. Koichi Ogawara Dr. Jun Takamatsu and Mr. Shunsuke Kudoh for teaching me many things. Without them, my knowledge would not be enough to do this research.

I am very grateful to Mr. Shinichiro Nakaoka who is the one that developed the simulation program for robot for spending so much time helping me.

Thanks to Mr. Takaaki Shiratori the very good friend who supports my life in Japan very well.

I am also very grateful to all the members of Humanoid Robotics Group of AIST and all members of Ikeuchi laboratory that support this research in various ways.

Contents

List of Figures

Chapter 1

Introduction

Humanoid robots are showing many features that draw attention to this field of research. Presently there are various kinds of research on humanoid robot, such as motion control, man-machine interface, visual and haptic feedback, artificial intelligent, and so on. Our research group focuses on motion control especially using robot off-line to preserve Japanese traditional dances, which are considered as intangible cultural assets. However, some of them are disappearing because of a lack of successors. Using humanoid robot also popularizes these national assets. At present Aizu-Bandaisan, which is the Japanese traditional dance from Fukushima prefecture, is used as test motion.

The dance Aizu-Bandaisan is very complicated and is not easily performed by humanoid robot with existing algorithms. If our algorithm can deal with such dance, it would be benefit to other kind of motion also, such as whole body operation of humanoid robot.

In this work, it will focus on upper body motion, to generate motion that preserve characteristic of human motion, namely joint angle and end-effector position, and meet the constraint of robot, namely joint's angle velocity acceleration and dynamic force needed to drive actuator with link. Optimization, one of the techniques that are used to transform the motion of human to the capabilities of the humanoid robot, is used to transform human motion by specifying an objective function that preserves the salient characteristics of the original motion, and a set of constraints that limits the motion to physical constraints of robot.

First, the motion is simulated by program developed by [20]. Then the robot HRP2 (Humanoid Research Project2) is used to test the result.

In this chapter, first the need of preservation of traditional dances is explained. Section 2 explains the necessity of motion control method. Then previous works are

shown in section 3. Section 4 introduces our method. Finally organization of this thesis is shown in section 5.

1.1 Preservation of traditional dances

There are a number of cultural heritages in the world. However, some of them are gradually lost due to many reasons, such as old age in the case of building structure or no successor in the case of tradition.

Recently, there has been an attempt to preserve these heritages by the use of computer technology. Once entered into a computer, these heritages can be recorded and may possibly be recreated in the future. Traditional dances are also one of the cases, the work that use computer to preserve dances are such as Nakamura [19] Hattori [9] Soga [26] and Yukawa [34].

There are many traditional dances in Japan, some of which have disappeared for the lack of successors. So one of the goals of this work is to preserve these dances by using the technology of computer vision and robotics.

One typical approach to preserve dances is to utilize a motion capturing system to capture human motion. This captured data enables us to observe dances from various viewpoints. Realizing a realistic movie of a performance by computer graphics is possible. However, just a movie replay is not very different from video recording. Captured motion data has potential for various applications. There are more advanced attempts, which enable applications beyond just a replay of a movie.

Using humanoid robot is one of such methods. It enables not just watching a dance movie but watching a dance by dancer. Also using humanoid robot seems to popularize these national assets.

1.2 The need of better motion control method

This part explains briefly another goal of this work. In order to use the robot for whole body on-line tele-operation or use it off-line as demonstrator, many techniques are used to transform the motion of human to the capabilities of the humanoid robot. The present works still have many problems, since the number of degrees of freedom (DOFs), range of joint motion, achievable joint velocity or acceleration, and dynamic motion capability of today's humanoid robots are far more limited than those of the average human subject.

If the planned motion does not meet the capabilities of robot, the error in joint angle trajectory would occur. Mostly the robot is set to stop after the input trajectory is different from the actual trajectory driven by controller for some certain threshold,

with the main objective to avoid collision that may cause severe damage to robot. This threshold for stop criterion is very low, such as 1 or 2 percent of joint moving range in case of the robot HRP2 that we use.

This means that if the planned motion has any conflict with the physical constraint of robot, and that this conflict results in error larger than threshold, the motion cannot be tested. One may ask that why not increase the stop criterion of the robot. The answer is that doing so can only solve the problem temporarily, for some motion.

Next the problems of existing methods are explained.

1.3 Previous work

Recently, Pollard [22] has realized a dance by a humanoid based on captured human motion, but their method shown problems due to the joint's angle and velocity limit algorithm. It has 2 drawbacks. Firstly, since each joint is scaled separately, the overall motion may be different from desired one, such as hand position is not correct. Secondly, though the joint angle and velocity are limited to the actuator capabilities, however actually robot has dynamic characteristic. It means that there are innertia of link and actuator, friction of gear, Corioris and centrifugal force, which all of these are altered by different motion. This dynamic characteristic may cause the capabilities of actuator to be different from that in specification, means that algorithm will not always work especially for abrupt motion.

For the optimization-based methods that minimize the objective function, such as the different between joint angle of robot and that of human, with a set of constraints that deals with the motion to physical limits of robot, such as velocity, there have been many works. Safonova [24], for example, performed optimization for each time frame of motion to preserve characteristic of human motion. There are 3 drawbacks in such kind of optimization. Firstly, though at each time frame, the motion is optimum, however the relation between time frames is not considered. This causes spike signal in the motion. Secondly, though at each time frame, the motion may be optimum for the selected objective function and constraint (of course the optimum motion would change if other objective function or constraint are selected), it is likely that it is not optimum in the sense of overall motion. For example, while doing optimization, if the joint trajectory at any time frame falls in the Gimbal lock [A2], at the next time frame it will not be able to flee from the lock position, though the trajectory is optimized. Thirdly, dynamic characteristic is not considered.

So the optimization-based method that exploits data representation technique is considered. Instead of optimizing joint angle for each time frame, the motion is

represented by some function. Then optimize the coefficient of such function. In other words, joint angle is viewed as determistic signal. This will ensure the smoothness and the overall motion will be optimum, in the sense of selected objective function. The examples of using this kind of optimization are Gleicher [7] Lee [14] and Ude [29]. [7] and [14] used this kind of optimization to retarget character, tested by computer graphic so it cannot be used in robot since the physical characteristics are not appropriate. [29] performed this optimization using with B-spline wavelets that can represent the detail motion well. However their algorithm didn't deal with velocity and acceleration limits effectively.

The main drawback the above previous works is that physical limits of robot are not considered. Though some works such as in [29], they used the objective function that minimize physical limits such as acceleration, however it is not guaranteed that such limits will be limited to the specification value. Moreover, again, the dynamic characteristic is not considered.

Furthermore, [20] already controlled robot to perform dances including leg motions and was able to maintain its balance independently, however the some of the physical limits of limbs are not considered. Also there are works about balance control such as Nishiwaki [21] Yamane [32] and Tak [27]. While the topics about walking or running robot can be found from Kajita [11] Endo [5] Nagasaki [18] and Yokoi [33]. So the issues of physical limits excluding balance of body are considered here.

1.4 Improvement in this work

More or less, the optimization-based method that exploit data representation technique seems to be good method, since the objective function can be selected to adjust hand position and many constraints can be set to meet the capabilities of robot, which is the drawback of [22], and it superior performing optimization for each time frame. Hence we focus on using this technique, and develop it to deal with physical limits of robot.

Instead of minimizing characteristics of motion by the objective function, it will be limited by the constraint of optimization. This would guarantee that optimized motion meets the physical limits of robot. Of course not only velocity and acceleration are considered, dynamic force is also exploited as one of constraints.

Among well-known data representation techniques, hierarchical B-spline is used since its hierarchical structure enabling refinement of motion to be more precise than just mere B-spline. Furthermore we have exploited B-spline's coefficient in velocity acceleration and dynamic force functions, so that it can be used in constraints

for optimization. It is also shown that limiting via hierarchical B-spline is easier than that of B-spline wavelets, which its function is more complicated.

We hope that our work will cope with all needs for retargeting upper body of humanoid robot, namely angle velocity acceleration force limit, predicting the number of B-spline control points in optimization so that the problem is not ill-conditioning nor the number of control points is not enough, and style preservation.

1.5 Organization of this thesis

In the next chapter, the system overview is shown. Then data representation methods are explained in chapter 3 with the criterions to choose method that match with user task. Also choosing between 2 well-known method, hierarchical B-spline and B-spline wavelets, is presented. Chapter 4 explains the optimization theory and algorithm that is used in this work. Then hierarchical B-spline based optimization is described in chapter 5. Chapter 6 gives the result of both simulation and experiment with the robot HRP2. Then the conclusion is in chapter 7. Finally there is appendix that shows the detail and disadvantage of inverse kinematics based optimization, and the Gimbal lock problem.

Chapter 2

Overview

This chapter describes an overview of retargeting process, transforming the motion of human to the capabilities of the humanoid robot, as shown in figure below.

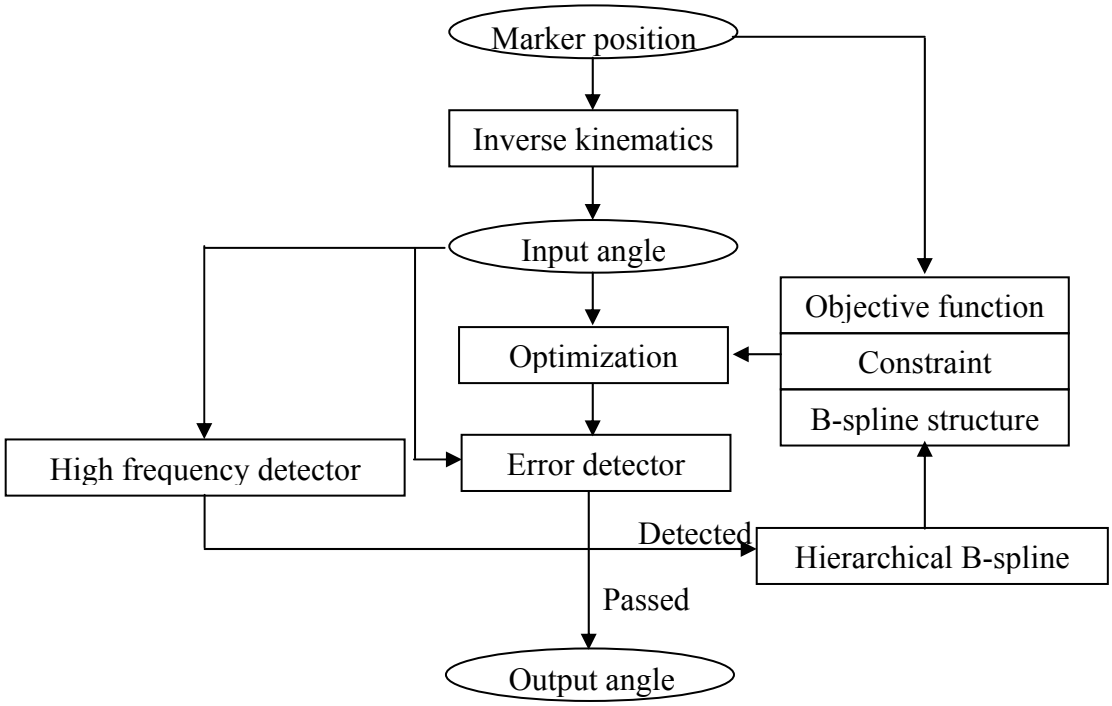

Figure 2.1 System overview

The process of data acquisition is shown in section 1. From input motion in the form of marker position, it is transformed to the joint angle data that is useable by robot, described in section 2. Then optimization is applied with B-spline data structure, briefly explained in section 3 and 4. Finally high frequency motion and error are checked. If there is any period of data that should be refined, hierarchical B-spline is applied, briefly described in section 5. Finally section 6 shows the robot platform.

2.1 Data acquisition

Aizu-Bandaisan is the Japanese traditional dance from Fukushima prefecture that is located in the northeastern part of Tokyo the capital city of Japan. Aizu is an area in Fukushima prefecture, while Bandaisan means the mountain Bandai. The selection of this dance is based on its importance as natural asset, and the convenience to capture the dance motions, which is done at the facility of Aizu University. Also its motion is complicated that challenges us more than just mere everyday life motion such as walking or pick up things, that are widely researched.

In order to acquire the motion data the motion capturing system, which can acquire time series properties of posture such as joint position or joint angle, is used. We use two popular types of the system, the magnetic method type and the optical method type. Both systems track positions of several markers in 3D space. Magnetic systems also track orientations of markers. Markers are attached to a performer and body motion of the performer is captured as sequence of each marker.

The magnetic system consists of a magnetic field generator and magnetic sensors as position markers. The sensors are attached to a human body and the acquired data are transmitted by radio. However the magnetic sensor is disturbed by metal object in the environment, which is a severe condition for using this system. In a room of a reinforced concrete building, precise data are difficult to acquire. Furthermore, the available capture are restricted within a small area because of the capacity of the magnetic field generator. Such as in our case, the magnetic system called Motionstar from Ascension is used with the translation range of only ± 10 feet. Although the system has above restrictions, it is useful because the accurate data, 0.6 inch RMS at 10 feet range in our case, is captured in real time under proper conditions. It is also has the merit of portability. Figure below shows the device of system.

Figure 2.2 Magnetic type motion capture system; Ascension

The optical system consists of several cameras and optically active markers and lights. In most systems, an infrared ray is used to capture markers. In that case, camera is infrared one, marker is made of material that reflects infrared ray well, and light is infrared ray. In order to distinguish marker clearly from the background, usually the human must wear a single-colored suit with the markers on it. Each camera has a different position and orientation from the others, which enable it to shoot markers that may be hidden from some cameras, and to calculate 3D position of markers by integrating movies obtained from different viewpoints with the principle of triangular surveying.

Though the optical system can acquire only the position of the marker, the system can have many markers to acquire the detailed motion as far as the markers can be distinguished. Also, the optical system capturing area can be wider than that of the magnetic system. On the other hand, the optical system may requires post processing by a human operator for calculation of the marker position if the occlusion happens in captured movie. Although the optical system has the disadvantage of the complex task, it does have the advantage of precision. ?

So the dance motion is acquired by optical motion capture system as shown in below figure. The system consists of 8 cameras with 33 markers. The capturing rate is set to 200 frames per second to match with the robot platform input rate.

Figure 2.3 Optical type motion capture system with professional dancer

Of course the occlusion of marker can happen, so the marker's entire trajectory is automatically interpolated by motion capture program. Although there are 33 markers on the body of dancer, only 28 of them are important markers for interpolating the trajectory, as shown in figure below.

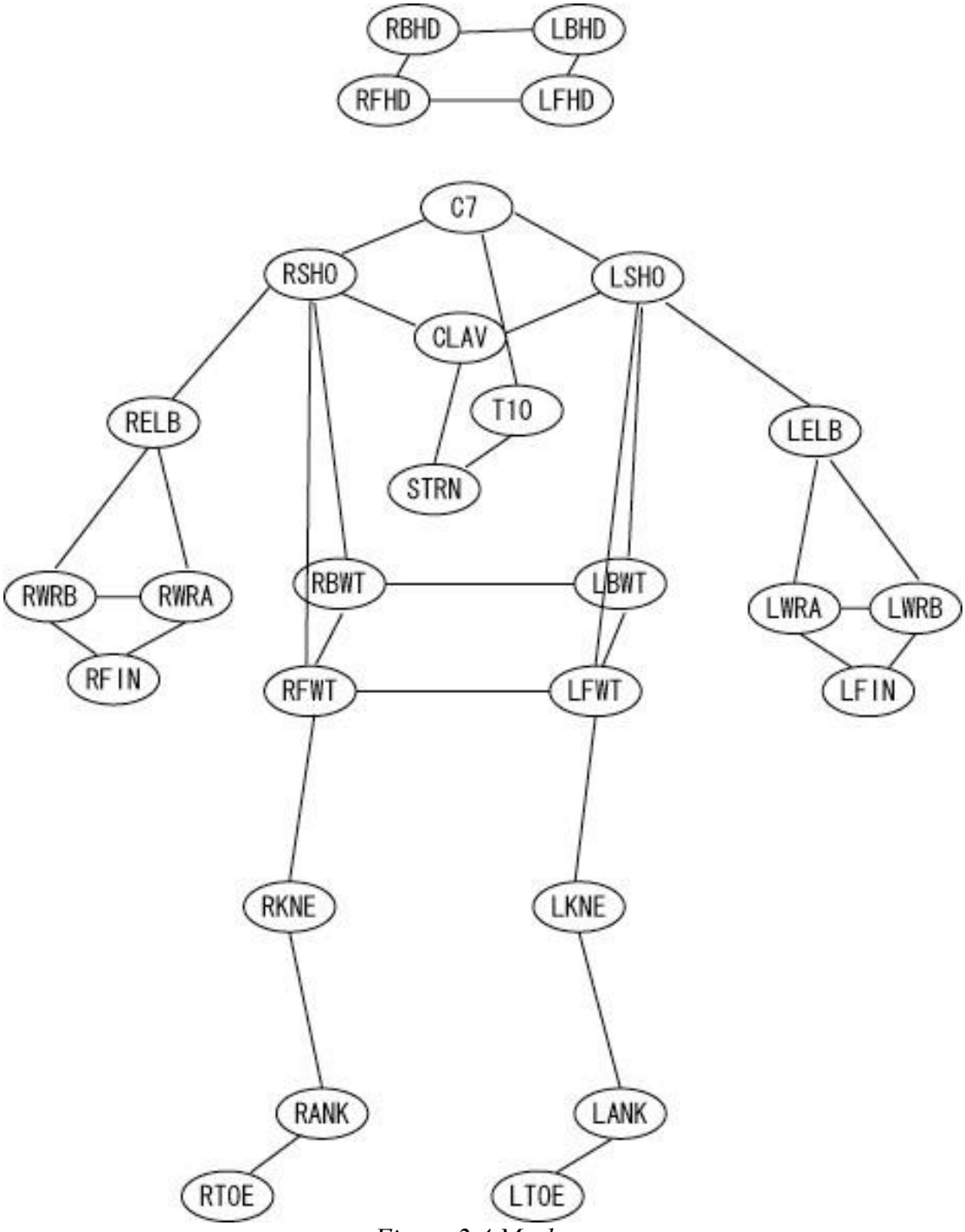

Figure 2.4 Markers

2.2 Data conversion

The dance data must be transformed to the form that is usable by the robot. Inverse kinematics is such method that calculate joint angle from position and orientation of link.

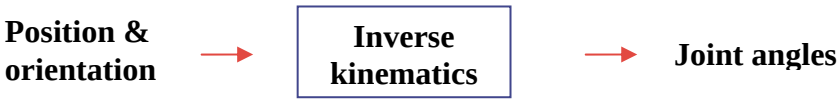

Figure 2.5 Inverse kinematics

If we define the coordinates of a manipulator as the *n*-dimensional vector of joint angles $Q \in Rn$, and the position and orientation of the manipulator's end-effector as the *m*-dimensional vector $S \in Rm$, the forward kinematics function can generally be written as:

$$S = f(Q) \tag{2.1}$$

While what we need is the inverse relationship:

$$Q = f^{-1}(\mathrm{S}) \tag{2.2}$$

There are various kinds of inverse kinematics. One of the well-known methods that uses Jacobian matrix can actually be exploited as optimization directly, without optimization routine. However we did not use it since drawbacks described in appendix [A1].

Since we have enough data from markers' position attached on dancer's cloth, geometrical inverse kinematics will be used as example of 2 links in 2 dimensions below.

Given 2 unit length links as shown in the figure below, the angle between can be calculated as below.

$$\begin{aligned} & a + b\cos\theta = x \\ & 1 + 1\cos\theta = 1.7071 \\ & \cos\theta = 0.7071 \\ & \theta = 45^0 \end{aligned} \tag{2.3}$$

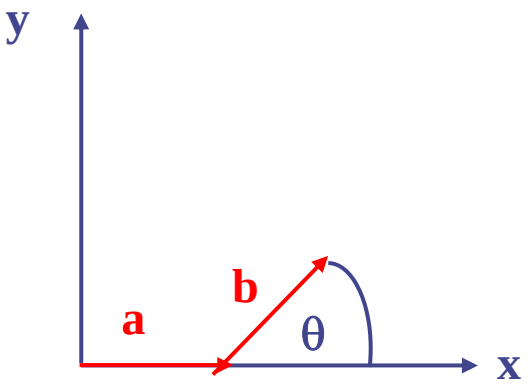

Figure 2.6 Illustration for geometrical inverse kinematics

2.3 Data representation

As optimization each time frame has drawbacks stated, namely the relation between time frames is not considered that spike signals occur in the motion and the result it is not optimum in the sense of overall motion, so each joint angle of a selected period would be represented by a set of B-spline coefficient, called control point. Then these control points would be optimized as explained in the next topic.

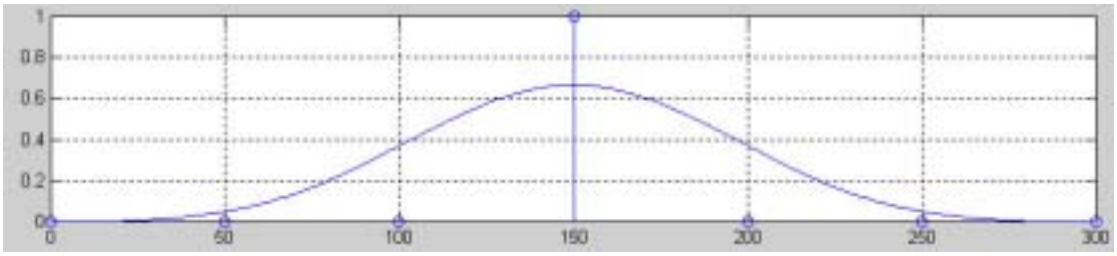

Figure 2.7 Example of a B-spline curve

Here is the background idea of using B-spline.

$$q(n,[p]) = B - spline(n, p_0, p_1, ..., p_{cp}) \tag{2.4}$$

where q is a discrete data of joint angle derived from B-spline function, $[p]$ is a set of control points p which is consisted of $cp + 1$ control points.

Actually the coefficient of B-spline is the only unknown for optimization. Also it will be exploited in physical limit constraints.

The detailed explanation of data representation and the criterion to choose one are explained in chapter 3.

2.4 Data optimization

Now the optimization is used to adjust B-spline control points so that the objective function is minimized.

$$[P]_{opt} = \underset{[P]}{\arg\min} \sum_{n=1}^{N} f(\Theta(n), Q(n,[P])) \tag{2.5}$$

where $[P]$ is the matrix of control points for all considered angles, N is the sampling length of a trajectory, $f()$ is the objective function, $\Theta(n)$ is the vector of joint angles derived from inverse kinematics $\theta(n)$, $Q(n,[P])$ is the vector of joint angles calculated from B-spline function $q(n,[p])$.

Usually we use objective function that minimize the difference between input joint angle and joint angle calculated from B-spline function, and the difference between hand position is also taken into account if the hand position is important, such as clapping period.

With the constraint that would limit some physical constraint of robot in the function $Q(n,[P])$, the optimization is applied.

The theory and algorithm of optimization are in chapter 4, while B-spline based optimization is explained in chapter 5.

2.5 Data refinement

Usually a set of B-spline has limitation in the sense that it cannot represent a curve that swings more frequently than some certain threshold, since the swing frequency depends on control point density. Note that frequency here does not actually mean frequency of signal. As a counter example, it can be seen that when the fourier transform is applied for spectrum, even a straight line has infinite frequency.

Even though the density of B-spline is appropriate, the error can occur due to the optimization process does not fully converge, which often happens because the stop criterion of optimization process is set before convergence is met, such as number of objective function called, etc.

Hence, to detect the error, the B-spline curve from optimization is compared to original angle.

$$e(n) = | \theta(n) - q(n,[p]_{opt}) | \tag{2.6}$$

where $e(n)$ is the error value.

Also, fortunately some error can be detected prior to optimization routine by detecting frequently swing pattern of each joint angle. Then the appropriate density of B-spline, in other words appropriate hierarchy, can be assigned to match with the swing frequency of motion.

Finally, if the error of any part of B-spline curve is larger than threshold or if the input joint angle swings more frequently than a certain threshold, it will be fed to hierarchical B-spline decomposition, to get new B-spline that has greater number of control points.

$$[hp] = (hp_0, hp_1, ..., hp_{hcp}) = Hie([p]_{opt}) = Hie(p_0, p_1, ..., p_{cp}) \tag{2.7}$$

where $[hp]$ is a set of $hcp + 1$ hierarchical control points hp and $hcp > cp$, Hie denotes the decomposition function.

Then this new set of control points, only at the period of joint angle that has error, will be optimized again to find a new optimum set of B-spline.

$$[HP]_{opt} = \underset{[HP]}{\arg\min} \sum_{n=1}^{N} f(\Theta(n) - Q(n,[HP])) \tag{2.8}$$

where $[HP]$ is the sets of hierarchical control points for all considered angles

Of course, B-spline can have many hierarchies. In our case of Aizu-Bandaisan traditional dance, decomposition of only one or two times is appropriate to represent the original motion well.

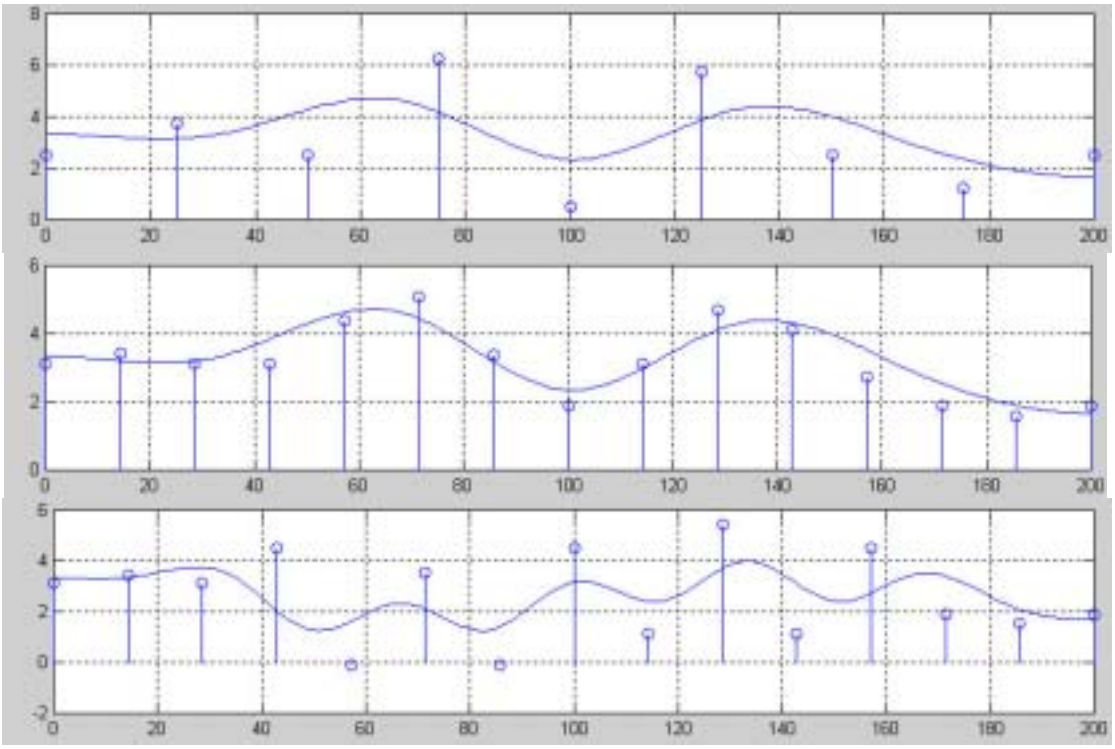

Figure 2.8 Altering control points of hierarchical B-spline
Above) A B-spline curve
Middle) B-spline curve with hierarchical structure
Below) Result of altering control points

It can be seen from above figures that after a set of B-spline control points is decomposed to higher hierarchy, it can represent the curve that has higher swing frequency.

Not only too few number of control points is the problem, but too many of them can also causes uneven output trajectory. This is one of the problem called ill condition in optimization. There has been researched on control point deletion such as Korb [13] or Lyche [17]. However this is not big problem in our work since frequently swing pattern is detected and the appropriate number of control points is assigned. In other words, new simple technique is developed in this work so the problem of too many control points is resolved.

The detail about hierarchical B-spline and how to detect the error before solving are in chapter 5.

2.6 Robot platform

The robot HRP2 that we use is the robot platform for the Humanoid Robotics Project headed by the Manufacturing Science and Technology Center (MSTC). The project is sponsored from the Ministry of Economy, Trade and Industry (METI) through New Energy and Industrial Technology Development Organization (NEDO). The total robotic system was designed and integrated by Kawada Industries, Inc. together with Humanoid Research Group of National Institute of Advanced Industrial Science and Technology (AIST). Yaskawa Electric Corporation provided the initial concept design for the arms, AIST 3D Vision Research Group and Shimizu Corporation provided the vision system, and Mr.Yutaka Izubuchi, a mechanical animation designer famous for his robots that appear in Japanese anime designed the appearance, as in figure below.

Figure 2.9 Humanoid robot HRP2

HRP2 is 154 cm height with the mass of 58 kg including batteries. It has 30 degrees of freedom (DOF) including two DOF for its hip. Its highly compact electrical system packaging allows it to forgo the commonly used "backpack" used on other humanoid robots. The detail about prototype robot HRP2P can be read from Kaneko [12].

Appealing things of this robot are such as it has slim body structure which benefits to collision avoidance, and the harmonic actuator used is powerful enough to represent human motion.

For harmonic actuator, the powerful motor that is often used in advanced humanoid robot, the main characteristic is the force (or torque) limit and the velocity limit as an example calculated from rated value shown below.

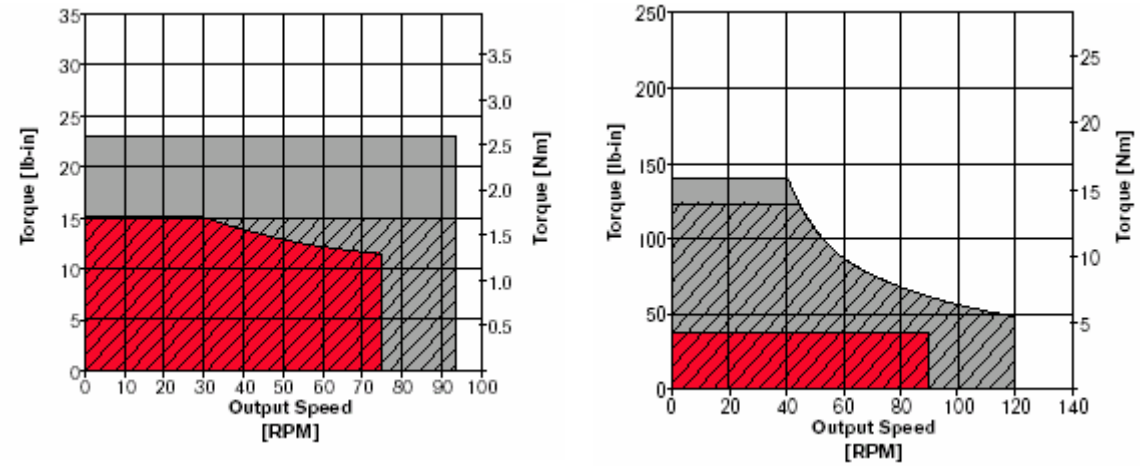

Figure 2.10 Torque-speed characteristic of harmonic actuators from Harmonic Drive Technologies. Red area is maximum continuous torque, gray area is peak instantaneous torque, and lined area is maximum repeated torque.

Thus, in addition to angle limit, both velocity and force should be taken into account as constraint for optimization. Note that in our case velocity limit is given from AIST while force limit is calculated by performing inverse dynamics to experimental data.

Chapter 3

Data Representation

As mentioned in the introduction, performing optimization to joint angle at each time frame results in jerk motion and the result may not be optimal in the sense of overall motion. Hence the joint angle trajectory will be represented by a set of continuous function's coefficients. Then instead of optimizing joint angles directly, the above coefficients will be altered so the function will be able to best represent the joint trajectory. In other words, in the field of signal processing, the joint trajectory is viewed as deterministic signal, the signal that can be represented by the function.

Consequently, one of the most important things about this work is selecting this basis function. The chosen function will play a very important role in the optimization routine. There are points that must be considered noted below.

Whether the function's characteristic can represent the joint trajectory well or not must be considered. It would not be clever to use any existing function to represent trajectory, such as although using a polynomial function with some high order maybe sufficient to represent a joint trajectory with some short length, but when it comes to longer trajectory or more complex trajectory the polynomial function may not have enough ability to represent such curve.

Next the direct relation in representing trajectory is also an issue. Such as although Fourier transform is the very powerful function to analysis the spectrum of signal, using discrete frequency domain coefficients to represent continuous joint trajectory is not good idea since there is no direct relation between these two domains. Their relation is quite complex in the sense that changing two coefficients in frequency domain have different results in trajectory domain depending on the frequency.

So the well-known spline function was proposed. Spline curves originate from flexible strips used to create smooth curves in traditional drafting applications. The

order of function can be selected to match with curve characteristic, and there is a quite direct relation between the curve and the coefficient called control point.

Note that there are many variants of spline, if only word spline is referred, it means conventional polynomial spline or natural spline that is constructed of piecewise polynomials that pass through a set of control points. Figure below shows the curve of 3rd, 2nd, and 1st order spline respectively. Note that, here, the 2nd and 1st order spline is calculated by differentiating the 3rd order curve.

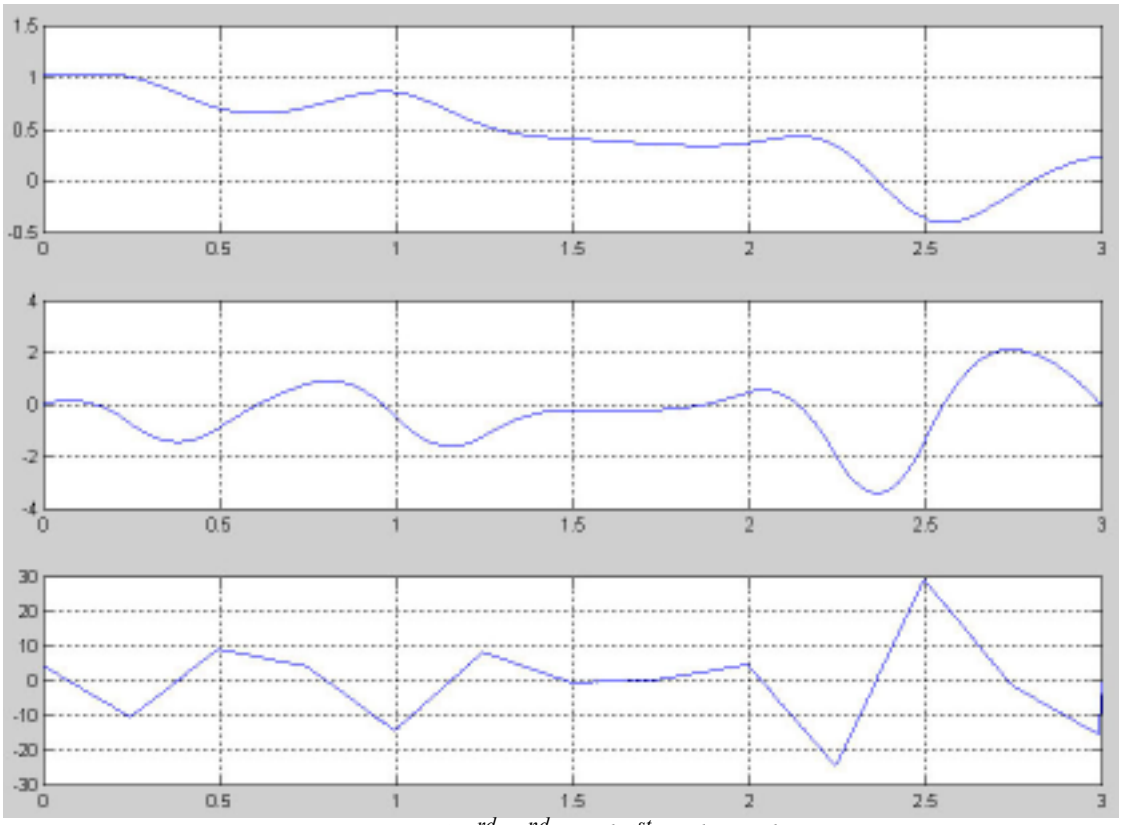

Figure 3.1 3rd, 2nd, and 1st order spline curve

However changing one coefficient in the spline function or polynomial function or frequency domain of Fourier affects the whole trajectory, which is not a good characteristic.

We need the function that has local property, changing a coefficient affects only a limit area. So the spline that has such property is introduced in section1. Then wavelet is also shown that it has such property in section2. Finally the comparison between them is considered in section3.

3.1 B-Spline

As stated above, conventional polynomial spline are not popular in curve representation or in CAD systems since they are not intuitive for iterative shape design. B-spline (sometimes, interpreted as basis spline) was investigated by a number of researchers in the 1940s. But B-spline did not gain popularity in industry until de Boor and Cox published their work in the early 1970s. Their recurrence formula to derive B-spline is still the most useful tool for computer implementation.

Let a vector known as the knot vector be defined

$$[t] = \{t_0, t_1, ..., t_{kv}\} \tag{3.1}$$

where [t] is a nondecreasing sequence with $t_i \in [0,1]$, and define control points

$$[p] = \{p_0, p_1, ..., p_{cp}\} \tag{3.2}$$

Define the degree as

$$dg = kv - cp - 1 \tag{3.3}$$

The knots $t_{dg+1}, ...t_{cp}$ are called internal knots.

Define the basis functions as

$$\begin{aligned} N_{i,0}(t) &= \begin{cases} 1 if\ t_i \le t < t_{i+1}\ and\ t_i < t_{i+1} \\ 0\ otherwise \end{cases} \\ N_{i,dg}(t) &= \frac{t - t_i}{t_{i+dg} - t_i} N_{i,dg-1}(t) + \frac{t_{i+dg+1} - t}{t_{i+dg+1} - t_{i+1}} N_{i+1,dg-1}(t) \end{aligned} \tag{3.4}$$

Then the curve is defined by

$$S(t) = \sum_{i=0}^{cp} p_i N_{i,dg}(t) \tag{3.5}$$

is a B-spline.

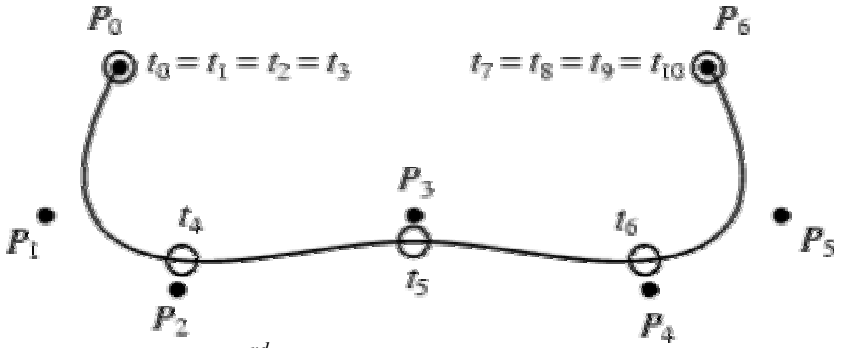

Figure 3.2 Example of 3rd order B-spline with 7 control points; MathWorld

Moreover since joint trajectory's velocity continuity is important so that the trajectory can be actually represented by the robot, and acceleration continuity is preferred in the senses that it would be benefit to controller tracking and prevent the actuator wear, so cubic B-spline or 3rd order B-spline is used.

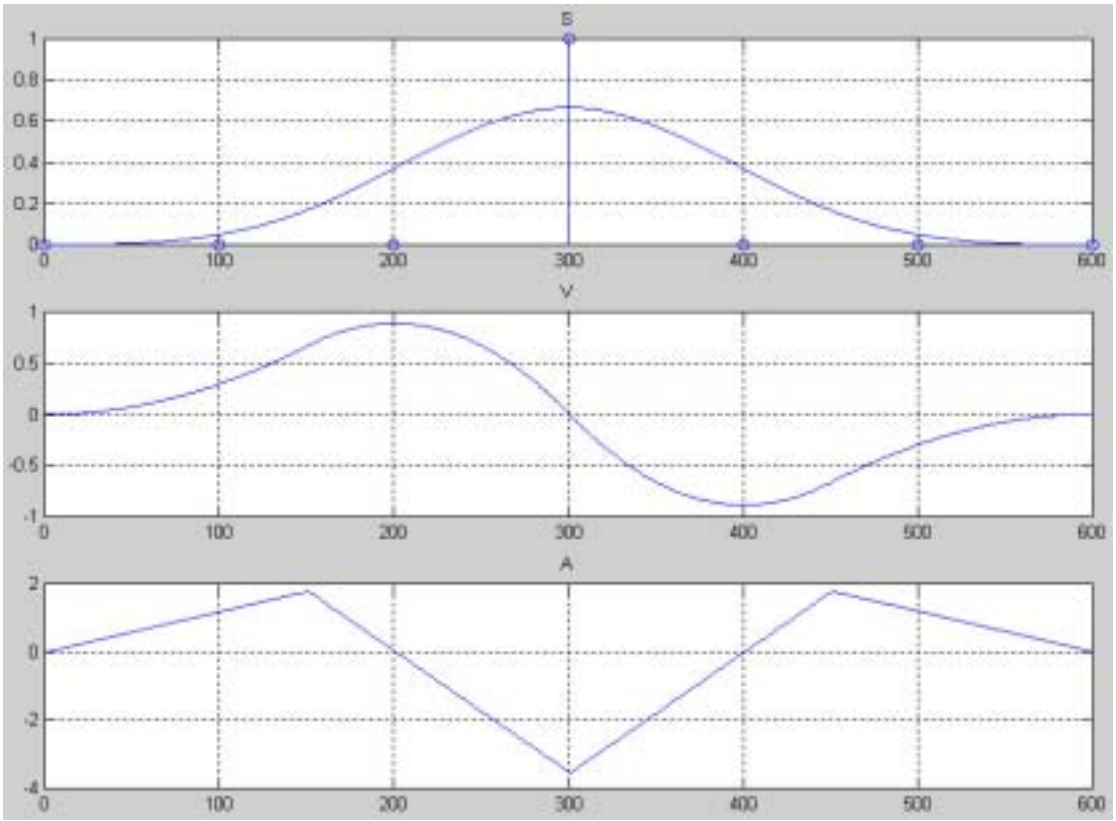

Figure 3.3 Cubic B-spline's scale, velocity, and acceleration

Although this illustration of B-spline is generated from 7 control points, only one of them has magnitude of 1 while other is set to be 0. Lower number of controls point cannot show the perfect shape of spline, because 3 control points at the beginning or the end of curve correspond to truncated basis. Since all control points has the same basis, though basis at the end is truncated, it is called periodic B-spline. Non-periodic has different shape of basis at the beginning or the end of curve. It can also be used in our work. However periodic one is chosen since it has same equation for any basis, is easier to be exploited as constraints for optimization.

As mentioned one of the reasons of using B-spline in our work is because of its local support property, changing a control point affects only a limited area of curve.

$$N_{i,dg}(t)\begin{cases} > 0 \, if \; t_i < t < t_{i+dg+1} \\ = 0 \, elsewhere \end{cases} \tag{3.6}$$

In other words, a change of the control point p_i affects the B-spline curve locally only for $t_i < t < t_{i+dg+1}$.

It is needed to be mentioned that the number of control points for cubic B-spline has to be equal or higher than 4. To undersand this let use an example of 3 control points, or *cp*=2, from equation (3.3):

$$\begin{aligned} kv &= dg + cp + 1 \\ kv &= 3 + 2 + 1 \\ kv &= 6 \end{aligned} \tag{3.7}$$

However if kv = 6, or the number of knots = 7, is not enough. As explained prior in this section, the knot vectors outside internal knots, or that situate at the same position, is:

$$t_0,...t_{dg} \; and \; t_{cp+1},...t_{dg+cp+1} \text{ or } t_0,...t_3 \; and \; t_{cp+1},...t_{3+cp+1} \tag{3.8}$$

in case of cubic B-spline.

The meaning is there will have to be 4 knots situated at the beginning and the end of curve. The number of knots below 8, or number of control points below 4, is not enough.

Here is an example of changing a control point of cubic B-spline in 3rd order curve and the consequence in 1st and 2nd derivative calculated from differentiation.

The magnitude of control point is reduced from red to green to blue lines respectively. It can be seen from the 2nd derivative curve that these changes affects only 4 knot periods.

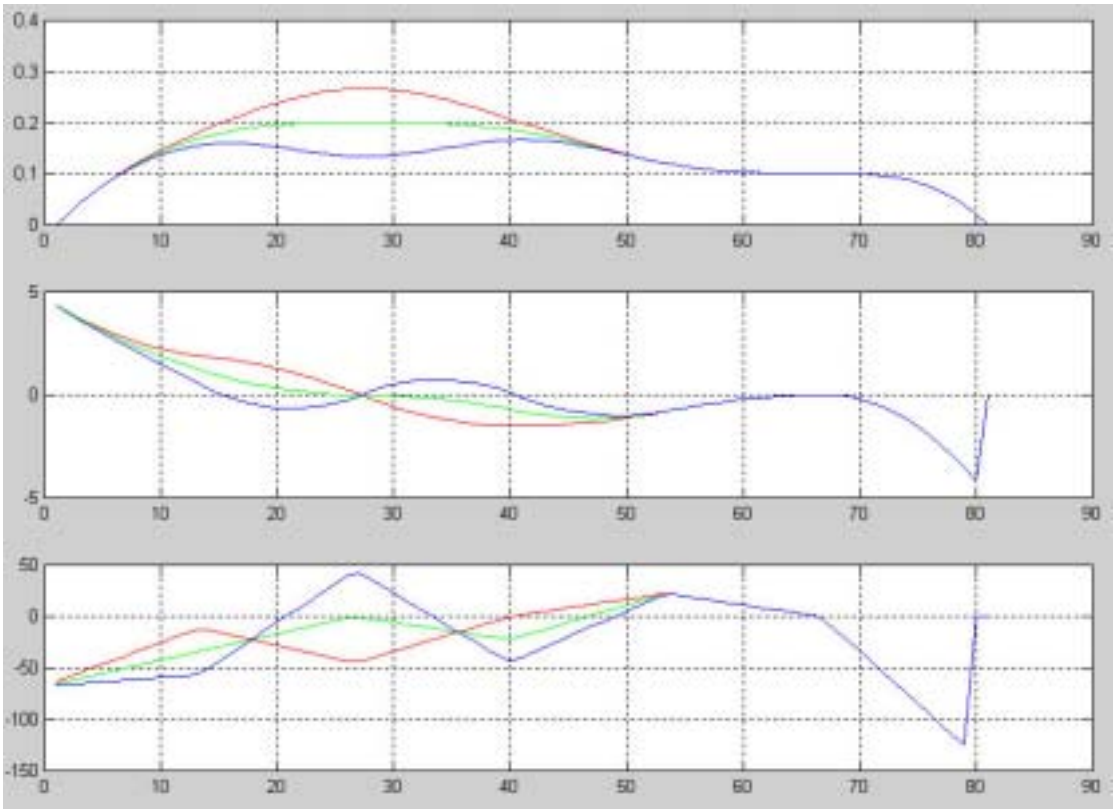

Figure 3.4 Cubic B-spline with 1 control point changed and its differentiated curves

Since control point p_i affects the cubic B-spline curve locally only for $t_i < t < t_{i+4}$, the period of $t_{i+3} < t < t_{i+4}$ will be controlled by only $p_i, p_{i+1}, p_{i+2}, p_{i+3}$. In other words, each knot period of cubic B-spline is functioned with 4 control points. So the equation for cubic B-spline will be solved based on this fact.

From recursive formula (3.4), the equation of cubic B-spline with 4 control points, p_0, p_1, p_2, p_3, is shown below. These 4 control points affect a period of knot vector, $t_3 < t < t_4$, as expected from equation (3.8).

$$S(t) = \frac{1}{6}\{(-t^3 + 3t^2 - 3t + 1)p_0 + (3t^3 - 6t^2 + 4)p_1 + (-3t^3 + 3t^2 + 3t + 1)p_2 + (t^3)p_3\} \quad (3.9)$$

In order to get the equation of velocity and acceleration for use as constraints for optimization, perform derivative with respect to time to the above equation, the velocity of cubic B-spline curve can be derived:

$$\dot{S}(t) = \frac{1}{6}\{(-3t^2 + 6t - 3)p_0 + (9t^2 - 12t)p_1 + (-9t^2 + 6t + 3)p_2 + (3t^2)p_3\} \quad (3.10)$$

Also the acceleration:

$$\ddot{S}(t) = \frac{1}{6}\{(-6t + 6)p_0 + (18t - 12)p_1 + (-18t + 6)p_2 + (6t)p_3\} \quad (3.11)$$

At the beginning of period, $t = 0$:

$$\dot{S}(0) = \frac{p_2 - p_0}{2} \quad (3.12)$$

$$\ddot{S}(0) = p_2 - 2p_1 + p_0 \quad (3.13)$$

At the middle of period, $t = 0.5$:

$$\dot{S}(0.5) = \frac{p_3 + 5p_2 - 5p_1 - p_0}{8} \quad (3.14)$$

$$\ddot{S}(0.5) = \frac{p_3 - p_2 - p_1 + p_0}{2} \quad (3.15)$$

It can be generalized in case of cubic B-spline for any number of control points that velocity and acceleration of each period is.

$$\dot{S}(begin) = \frac{p_2 - p_0}{2T} \quad (3.16)$$

$$\ddot{S}(begin) = \frac{p_2 - 2p_1 + p_0}{T^2} \quad (3.17)$$

$$\dot{S}(middle) = \frac{p_3 + 5p_2 - 5p_1 - p_0}{8T} \quad (3.18)$$

$$\ddot{S}(middle) = \frac{p_3 - p_2 - p_1 + p_0}{2T^2} \quad (3.19)$$

where T is the length of knot period.

This is another reason that B-spline is chosen as basis function. Instead of filtering velocity as in Pollard [22] or indirectly minimizing velocity acceleration jerk or energy as Ude [29] William II [31] or Rose [23], which may not yield physical limit of robot, the velocity and acceleration equations of B-spline at the beginning and the middle of knot vector are solved for use as constraints for optimization. Also it can be used as variable for inverse dynamic equation that has three inputs, joint angle, velocity, and acceleration, so that the equation can be exploited as dynamic force constraint. The detailed will be desctribed in chapter 5.

3.2 Wavelet

Originally, wavelet is used as frequency analysis application. It possesses time-frequency localization property, detecting a specific range of frequency at a specific period of time, such as analysis of non-stationary signals and real-time signal processing.

In order to understand wavelet, first, consider Fourier transform that exploits complex exponential term as basis function. It is widely used as frequency analysis tool. However Fourier does not have the characteristic of time-frequency localization. When one needs to detect a specific range of frequency, it is needed to input whole time series of signal, so detecting frequency at a specific time is not possible or is done by truncating the signal first which is not effective in some cases.

So in time-frequency localization application, Gabor transform was introduced. Its basis function is Gaussian, which has unique characteristic that it has the same shape in both time and frequency domain.

$$Gau_{\alpha}(t) = \frac{1}{2\sqrt{\pi\alpha}} e^{\frac{-t^2}{4\alpha}} \tag{3.20}$$

where $\alpha>0$ is fixed as window function.

Then the Gabor transform is:

$$(Gab_b^{\alpha} f)(\omega) = \int_{-\infty}^{\infty} (e^{-i\omega t} f(t)) Gau_{\alpha}(t-b) dt \tag{3.21}$$

that is, it localizes the Fourier transform of f around $t = b$.

However the width of the basis, or window, is unchanged for observing the spectrum at all frequencies. That this restricts the application of Gabor transform to study signals with unusually high and low frequencies.

Hence the wavelet transform was introduced. The idea is that other basis functions may also be used as window functions. First define space $L^2(\Re)$ of measurable function f defined on the real line $\Re$:

$$\int_{-\infty}^{\infty} |f(t)|^2 \, dt < \infty \tag{3.22}$$

For a non-trivial function $Wav(t) \in L^2(\Re)$, to qualify as a window function, it must satisfy the requirement that:

$$tWav(t) \in L^2(\Re) \tag{3.23}$$

The most important property not possessed by the Gabor transform is the additional condition:

$$\int_{-\infty}^{\infty} Wav(t)dt = 0 \tag{3.24}$$

which makes the shape of wavelet has both positive and negative parts.

Here is an example of a wavelet generated from 11 B-spline by Cohen [2] using the weight sequence of:

$$\frac{1}{256}\{5,20,1,-96,-70,280,-70,-96,1,20,5\} \tag{3.25}$$

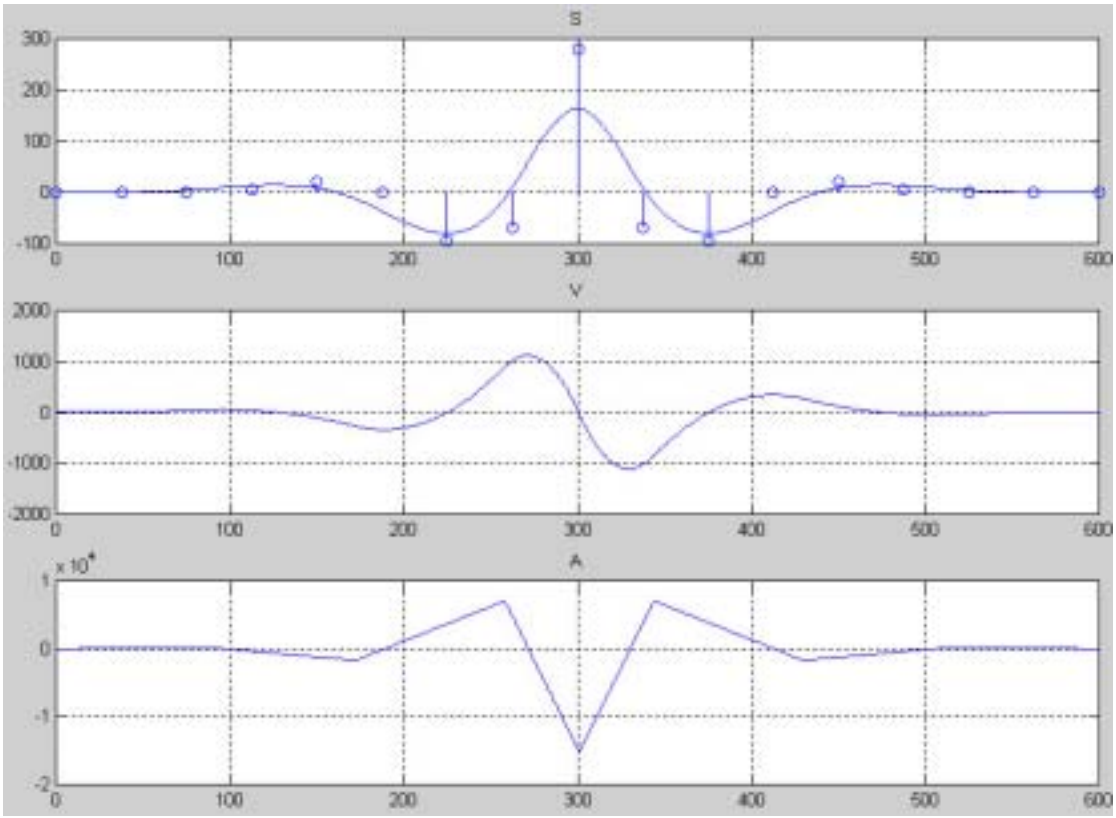

Figure 3.5 Wavelet's scale, velocity, and acceleration

Wavelets are not only benefit in frequency analysis field. As basis function, it can be used to represent a sequence of data such as a curve. Thus it can be used to represent motion in computer graphic field or for humanoid robot.

3.3 Choosing between hierarchical B-spline or B-spline wavelets

In order to discuss the advantages or disadvantages of each method, it should be noted that using just mere B-spline or wavelets often cannot effectively represent the trajectory of joint angle since the detail of basis may not be enough for complex motion. So, normally, B-spline is used in hierarchical structure that is using finer B-spline to represent detailed motion that cannot be represented in lower hierarchies. Also using this kind of hierarchical structure for wavelets is possible.

However, Gotler [8] suggested that using B-spline as lowest hierarchy and then use wavelets in higher hierarchies would be faster than using just hierarchical wavelets, and it is widely called B-spline wavelets. In the field of motion retargeting from human to humanoid robot, Ude [29] used B-spline wavelets to represent the trajectory of humanoid robot. Their work deals with physical constraints of robot indirectly, by minimizing not limiting the value, so it is not guaranteed that output angle from the algorithm can be represented by real robot. In other words, using wavelets or B-spline wavelets have disadvantages in the controlling of joint's velocity or acceleration. Consequently, moreover, the dynamic force equation that consists of velocity and acceleration term also has problem in case of using a basis that is a variant of wavelets.

Here are some discussions based on comparing between B-spline and wavelets. Reader should find it directly related to comparing between hierarchical B-spline and B-spline wavelets, two widely used methods.

Convergence when the same number of control points provided

For other wavelets that are (near) symmetric which are often used as trajectory representation, based on stated properties, the shape is similar to the one presented here, which has more peaks compared to B-spline in figure 3.3. As can be expected from such shape, using wavelets comparing to using B-spline in representing curve with the same number of control points, wavelets converge faster with less error, [8]. However with enough constraints are present, the wavelet no longer offer an advantage over B-spline. In our case, it is not necessary to test whether constraints are enough or not, since constraints are provided as physical limits of robot. So we would like to skip this comparison.

Number of control points per hierarchy

Another advantage of wavelet is that at each hierarchy, the number of basis elements of B-spline is roughly twice of that of wavelet. Thus using wavelets would reduce the optimization time from that of using B-spline as basis function. We accept this drawback but would like to note that most of retargeting works nowadays are done off-line, so this ratio of about 2 might not be the problem in the real world.

From the above 2 discussions, it might be more appropriate to use wavelet as basis function to retarget human motion in the field of computer graphic. However in optimizing human motion to humanoid robot, the critical conditions are limiting motion to capabilities of robot.

Physical constraints of robot

The primitive physical constraints of manipulator are angle, velocity, acceleration, and force. Since angle can be limited directly by limiting the magnitude of control points, and force can be derived from angle velocity and acceleration, as will be shown in chapter 5, only the velocity and acceleration are considered here.

From below 2 figures, it can be seen that a cubic B-spline has 2 peaks in velocity domain and 3 peaks in acceleration domain while a wavelet has 6 and 7 peaks respectively, that the complexity is about 3 times higher. Even if the small peaks in the case of wavelet are neglected there will be 4 and 5 peaks in both domains respectively, that the complexity is about 2 times higher.

Looking from the other side, instead of controlling the peak value, changing a cubic B-spline affects 3 knots while changing such wavelets affects 5 knots apparently (actually it is 7 in this wavelet), note that 2 periods in the curve is viewed as a period in the case of wavelet. It has to be chosen which at least one knot will be limited per changing a basis. The choice that has appropriate sensitivity could be 1st knot in case of B-spline and 3rd knot in case of wavelet. It can be seen that changing a B-spline affects only the future periods, while doing so to a wavelet affects both past and future periods. This makes velocity, acceleration, or dynamic force limits of wavelets ineffective or useless compared to that of B-spline.

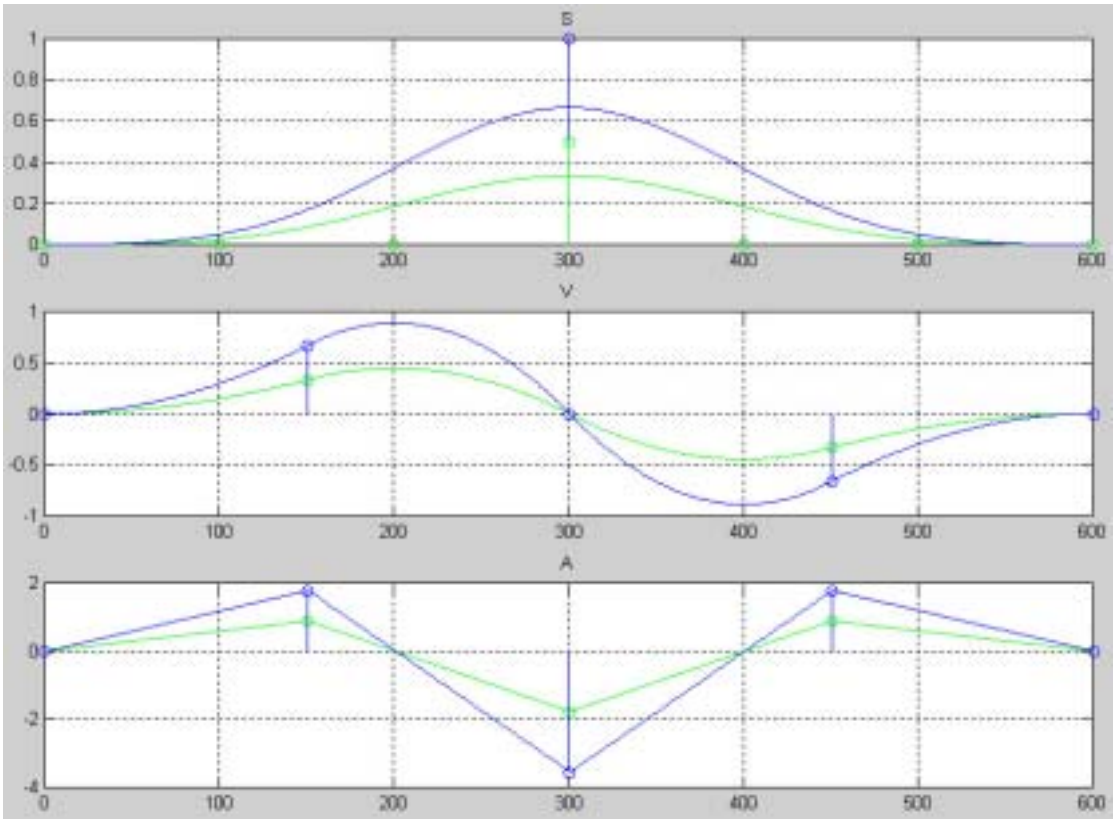

Figure 3.6 Changing a cubic B-spline, and the result in velocity and acceleration

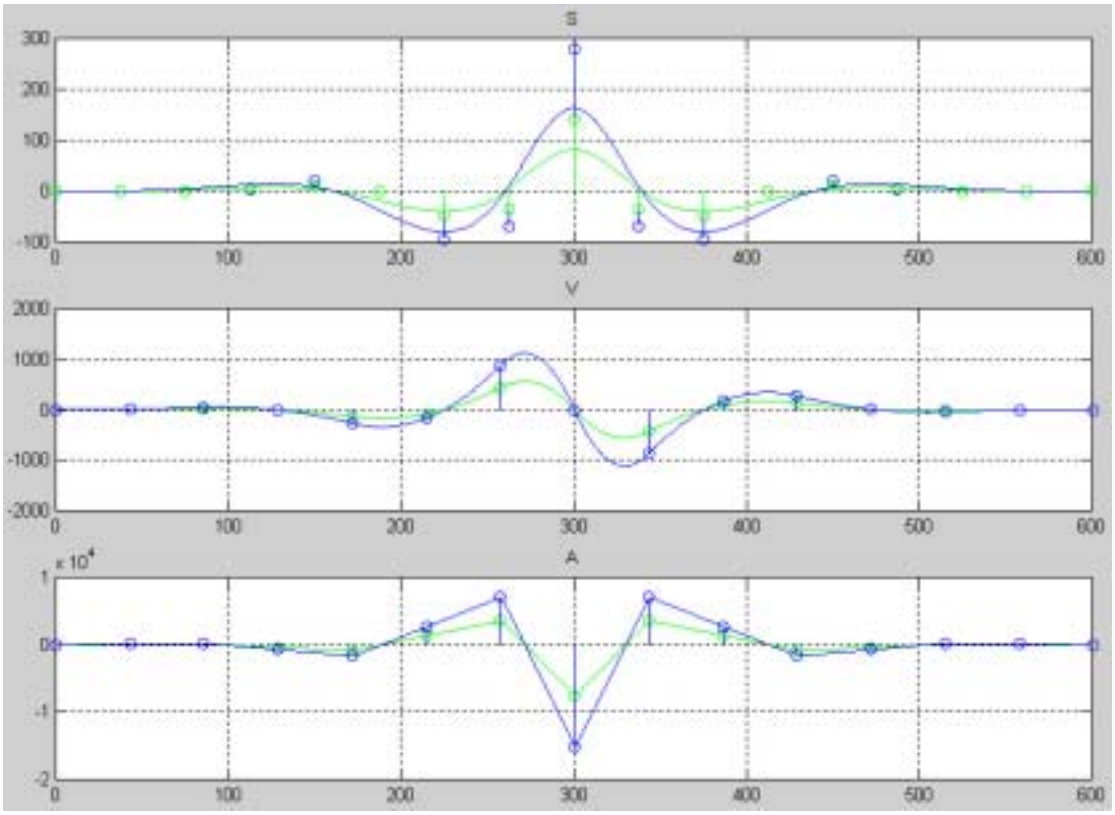

Figure 3.7 Changing a wavelet, and the result in velocity and acceleration

For velocity acceleration or dynamic force limits, it will be discussed in chapter 5, based on the equation for velocity and acceleration at the beginning and the middle of period solved previously in this chapter. Actually the checking for acceleration and force only at knot period is sufficient while it requires checking at and between periods in the case of velocity.

User interface

In many cases, user may need to adjust the curve after optimization, such as if the collision happen and so on. For this case, since adjusting a B-spline responds in a bell shape while a wavelet responds in a wave shape, manipulating wavelet basis does not provide a good interface.

From the above discussions, hierarchical B-spline is selected to represent the joint angle trajectory.

It will be shown in chapter 5 how to limit the values in scale, velocity, acceleration, and force domain as constraints for optimization. Also how to use hierarchical structure is explained. The next chapter first gives guideline of optimization theory and algorithm.

Chapter 4

Optimization Theory and Algorithm

This chapter describes the theory and algorithm of optimization, which is the problem of finding the best set of coefficients or function that is capable to reduce the value of certain function.

First local optimization is explained in section 1 for finding a local minimum set of coefficients or function. However the problem of untrue minimum can happen from such optimization, if the initial condition is not appropriate and the system has many minima. This may result in non-optimal result as shown in figure below. Also there can be the problem of Gimbal lock, depends on the objective function. The detail of Gimbal lock is in [A2].

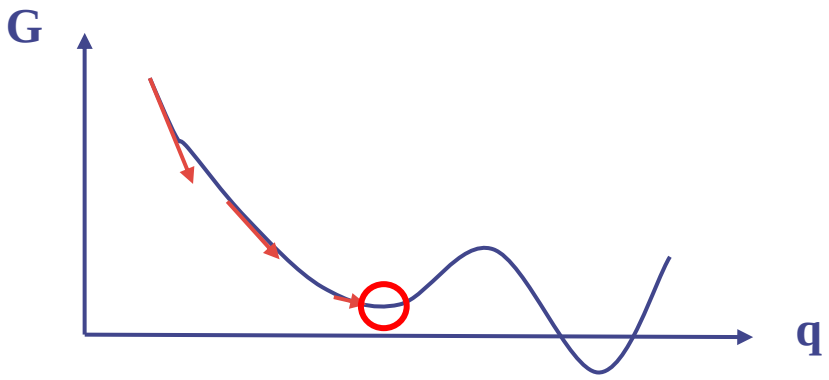

Figure 4.1 Problem of local optimization

A widely used algorithm of local optimization is shown in section 2. Although this algorithm is not used directly in our work, it will be the base algorithm for global optimization that is used here.

So global optimization, the method that capable to find best point in the system that has more than one minimal point, is introduced in section 3 to solve the problems of local one, because it can find best minimum under certain condition in the system that has many minima. Along with suitable data representation method, it is capable to not only find best minimum but also avoid Gimbal lock, so it is applied in our work. The algorithm of program from [10], used in our work, is introduced in the last section.

Actually only local optimization would be sufficient for our system at the present, which the objective function does not have complex form with many minima, nor Gimbal lock would happen, see the next chapter for objective function and [A2] for Gimbal lock problem. However we plan for the future that more complex objective functions with many minima may be used. Furthermore some of them may have the problem of Gimbal lock.

Before going to detail explanation, it is needed to mention that the word local or global optimization has 2 meanings and we have chosen one of them here.

The first one widely referred in the field of mathematic is that:

Local optimization:

Suppose $x^* \in S$ and that there exist an $\varepsilon > 0$ such that

$$f(x) \geq f(x^*), \textit{for all } x \in S : \| x - x^* \| < \varepsilon \tag{4.1}$$

then, x^* is a local minimum

Global optimization:

Suppose $x^* \in S$ and

$$f(x) \geq f(x^*), \textit{for all } x \in S \tag{4.2}$$

then, x^* is a global minimum

Another meaning is widely referred in the field of curve optimization, or trajectory optimization.

Local optimization:

At a round of routine, optimizing a (time) sample of curve, under certain objective function (and constraint).

Global optimization:

At a round of routine, optimizing the entire curve by representing curve with some appropriate function and then adjust the coefficient of such function, under certain objective function (and constraint).

Here, for the sake of explanation, we would like to use the notation of mathematician to define optimization.

4.1 Local optimization

From the above definition local optimization can be solved mathematically by finding a function x, on an interval, minimizing an integral functional of the form:

$$G = \int_{t_1}^{t_2} f[x(t), \dot{x}(t), t]dt \tag{4.3}$$

It can be proved that minima x must satisfy the Euler-Lagrange equation:

$$\frac{\partial f}{\partial x} - \frac{d}{dt}(\frac{\partial f}{\partial \dot{x}}) = 0 \tag{4.4}$$

Mostly, the objective function has the form as:

$$G = \int_{t_1}^{t_2} f[x(t)]dt \tag{4.5}$$

Or even

$$G = f[x] \tag{4.6}$$

The term $\frac{\partial f}{\partial \dot{x}}$ is 0 in the former case, while in the latter case it can be derived directly from gradient operation, that in both cases, equation (4.4) is reduced to:

$$\frac{\partial f}{\partial x} = 0 \tag{4.7}$$

4.2 Local optimization algorithm

Since finding the gradient of function explicitly is not practical in real world, local optimization can be solved numerically by using many algorithms. Sequential quadratic programming (SQP) is one of such algorithm.

From Taylor series expansion of a real function $f(x)$ about a point $x = x_0$:

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + ... + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n + ... \quad (4.8)$$

If it is to be approximated by only the 1st and 2nd derivative:

$$f(x) = f(x_0) + f'(a)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 \quad (4.9)$$

In the case of objective function $G(X)$ about a multidimensional point $X = X_0$:

$$G(X) = G(X_0) + \nabla G(X_0)^T (X - X_0) + \frac{1}{2}(X - X_0)^T \nabla^2 G(X_0)(X - X_0) \quad (4.10)$$

where ∇ and ∇^2 is Gradient and Laplacian operator respectively.

This means from the information at a start point, function's value, Gradient, and Laplacian, can be used to calculate the value of function at new point $G(X)$. Then it can be compared whether the new point has lower function value than the start point or not. If so, the new point is updated to be the new start point. The algorithm is iterated until there is no better new point is found.

However, this algorithm as well as other local optimization based algorithm is potentially affected by Gimbal lock problem, since Gradient and Laplacian at that posture doesn't enable robot to move as in [A2]. Thus, global optimization is introduced.

Note that global optimization algorithm that will be explained later is based on variant of SQP in local search routine.

4.3 Global optimization

Global optimization is the method that capable to find best point in the system that has more than one minimal point. Theoretically it can be explained as finding minimal point of nonconvex function.

To be able to understand the concept of nonconvex function, first let's define:

Convex set:

Set *S* is convex if the closed line segment joining any two points x_1 and x_2 of *S*,

$$(1-\lambda)x_1 + \lambda x_2 \tag{4.11}$$

belongs to the set *S* for each $0 \le \lambda \le 1$.

Let $f(x)$ be a real valued function defined on *S*.

Convex function:

$$f((1-\lambda)x_1 + \lambda x_2) \le (1-\lambda) f(x_1) + \lambda f(x_2) \tag{4.12}$$

Concave function:

$$f((1-\lambda)x_1 + \lambda x_2) \ge (1-\lambda) f(x_1) + \lambda f(x_2) \tag{4.13}$$

The above 2 equations are called Jensen's inequality.

A general nonconvex function contains convex and concave parts and as a result it exhibits multiple minima and maxima. A concave function defined within box constraints on all variables is a special case of a nonconvex function since it has a single maximum but several minima that correspond to the vertex points of the feasible region. Easily speaking, an example of nonconvex function is shown in figure 4.1.

4.4 Global optimization algorithm

There are various algorithms used to find true minimum. The effective and widely used ones are list below for reader to compare the usage.

First is the simplex algorithm. The idea is as simple as its name, begins by performing global searches for likely-to-be minimal points. Then at these points, the local optimizations are applied to move closer to minima. Finally the selects the best result.

Other 2 well known algorithms are simulated annealing and genetic algorithm. Simulated annealing is an algorithm that simulates annealing process of metal. Genetic algorithm exploits genetic diversity and natural selection process. Both algorithms involve a stochastic element, and they converge faster than simplex algorithm.

Simplex algorithm suits with constrained system with low dimensions, while the other two can deal with unconstrained system with high dimensions. In our case, low DOFs robot arm with constraint such as joint angle limit, simplex algorithm style method is selected with the variant of SQP as local optimizer. Next global search procedure is briefly explained.

The optimization package [10] that we are using is called multilevel coordinate search (MCS). It finds minima by splitting the search space into smaller boxes. Each box's area is defined by base point, whose function value is known. The partitioning is not uniform but is done where low function values are expected to be found. The illustration of splitting is shown by example of 2D in figure below.

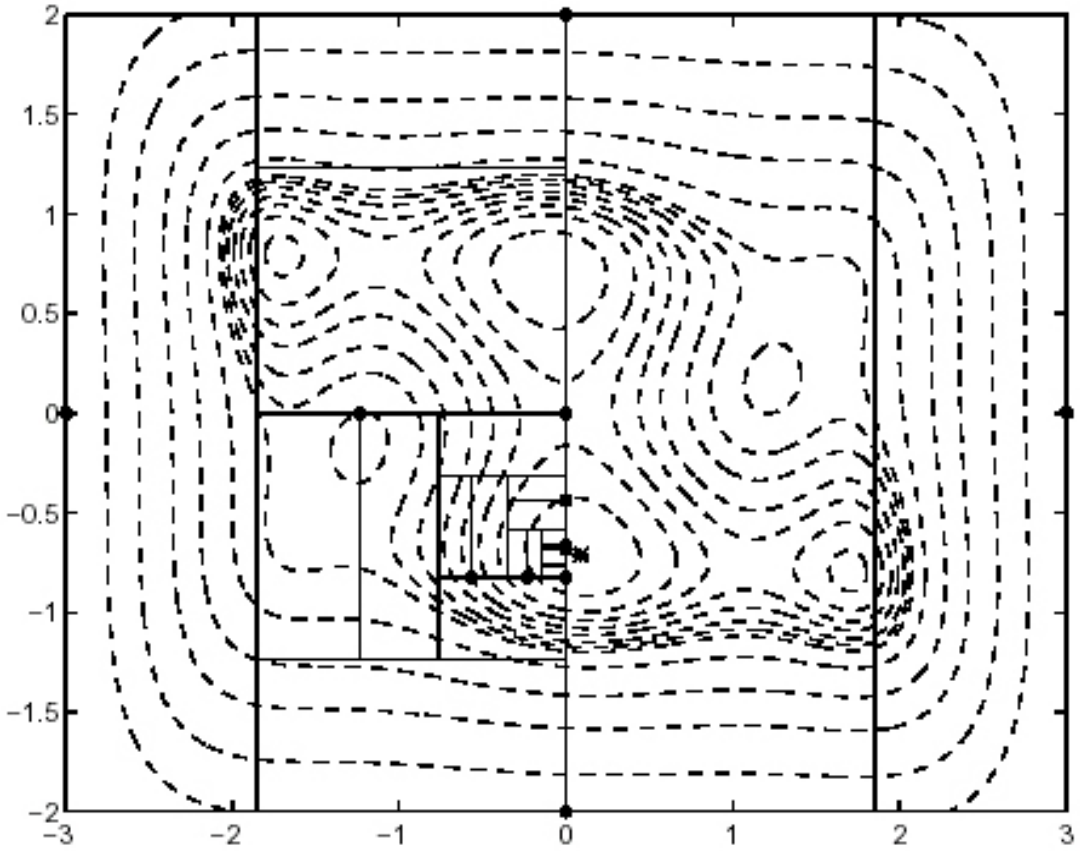

Figure 4.2 Splitting of MCS; [10]

As mentioned before this algorithm combines global search, splitting boxes with large unexplored territory, and local search by the variant of SQP, splitting boxes with good functional values. A level is assigned for each box, to state the number of times that box has been processed. The system is set to have limit level as one of stop criterion.

This optimization method can be used in two ways, with or without local search.

Without local search, it put base points of level max into the array of local minimums. Then the best point is selected to be the global minimum.

With local search, to accelerate the convergence at level max, the variant of SQP is performed again to the points before putting them into the array of local minimums. Precisely speaking, it is checked whether the base point of new box at level max is likely to be in the basin of local minimum already in the array of local

minima or not. If not, build a local quadratic model that called triple search, then defining a promising search direction by minimizing the quadratic model on a suitable box, and making a line search along the direction. Finally the best point is selected to be the global minimum.

Note this algorithm is adapted so that it has not only the bound constraint but also the constraints that suit with velocity, acceleration, or dynamic force limit.

Next, using this algorithm, B-spline optimization is explained.

Chapter 5

Hierarchical B-spline based Optimization

As the characteristic of appropriate data representation technique is considered in chapter 3, and B-spline is chosen. Now, how to use this kind of data representation in optimization, both in objective functions and constraints, is explained here. Then how to refine the optimized B-spline trajectory, which may still contain error, is explained next.

First let's mention again that a trajectory to be optimized is represented by B-spline function. Hence the coefficient of B-spline is the unknown for optimization. Then the optimization is used to adjust B-spline control points that the objective function is minimized.

Section 1 describes the objective functions that are used in this work. The selection of them is based on our goal to preserve salient characteristic of human motion.

Obviously, using only the objective function is not enough. The result trajectory may not be usable by the humanoid robot, since the physical constraints of robot. Many previous works chose to limit certain physical characteristics by minimizing them via objective function such as [29]. However doing so can only reduce the value, not restrict it, so such method may not give the trajectory that meets the capabilities of robot.

We have formulated many important physical limits of robot in the form of B-spline coefficient, so that the process of limiting them can be done directly in the constraint routine for optimization. That the optimized trajectory would meet with capabilities of robot.

Section 2 describes constraints for optimization that are derived in the form of B-spline coefficient.

Next sections explain the data refinement process.

Optimized trajectory can have error. The first reason is due to the optimization process does not fully converge, which often happens because the stop criterion of optimization process is set before convergence is met, such as number of objective function called, etc.

Hence, in order to detect the error, the optimized B-spline curve is compared to original angle to detect error, the same way as many previous works did.

Section 3 describes the process of error detector.

The second reason that error occurs is a set of B-spline has limitation in the sense that it cannot represent a curve that swings more frequently than some certain threshold, since the swing frequency depends on control point density. Also using too dense B-spline can result in over optimized B-spline curve, which swings more frequently than original curve.

We have found that detecting the frequency of each joint angle and assigning appropriate density of B-spline, in other words appropriate hierarchy, would help solving the density problems.

Section 4 explains the density detector.

Finally, if the error of any part of B-spline curve is larger than threshold or if the input joint angle swings more frequently than a certain threshold that cannot be represent by the current B-spline, which are detected from section 3 and 4 respectively, that set of B-spline coefficients will be fed to hierarchical B-spline decomposition, to get new B-spline that has greater number of control points.

Then this new set of control points, only at the period of joint angle that has problem, will be optimized again to find a new optimum set of B-spline.

Section 5 gives the detail about hierarchical B-spline.

Finally, the adaptations of constraints used for hierarchical B-spline are in the last section.

5.1 Objective function

The most important characteristic of human motion is probably the joint angle. Whether the everyday life activities or dancing, if the humanoid robot is to represent such motion, it is a good choice to use joint angle as data. Furthermore for some motions such as ones that interact with environment such as touching or those interact with own body such as clapping, the position (and orientation) of limb is also important.

The same goes for Japanese traditional dance Aizu-Bandaisan. In order to preserve the upper body motion of this dance, to pass through from generation to generation, it is vital for dancer to preserve the joint angles and hand position.

For the humanoid robot to represent the salient characteristic of dancing, it is found from experiment that mostly preserving only the joint angles is sufficient. So the joint angles of an arm after inverse kinematics is used as target trajectory.

$$f([P]) = \sum_{n=1}^{N} || \Theta(n) - Q(n,[P]) ||_2 \qquad (5.1)$$

where $[P]$ is the matrix of control points for all considered angles, N is the sampling length of a trajectory, $f()$ is the objective function, $\Theta(n)$ is the vector of joint angles derived from inverse kinematics $\theta(n)$, $Q(n,[P])$ is the vector of joint angles calculated from B-spline function $q(n,[p])$.

However for some motion in this dance such as put hands together (not touch), only comparing joint angles is insufficient because the size of robot's arm is different from dancer's arm. Furthermore, since the markers that are attached to the body of dancer via elastic suit may not balance, slip during capturing, or even occluded that the position sequences are derived from interpolation, so the marker position input to the inverse kinematics can have error. This error makes the joint angle(s) after inverse kinematics has a small different from the value it should be. However this little error in a joint angle may result in apparent error in limb position.

In these cases instead of using single objective function, another objective function that consider both joint angles and hand position is used:

$$f([P]) = \sum_{n=1}^{N} || Q(n,[P]) - \Theta(n) ||_2 + w || F.K.(Q(n,[P])) - Pos ||_2 \qquad (5.2)$$

where w is weight, $F.K.()$ is forward kinematics, Pos is hand position of human.

The forward kinematics method used here comprises of simple homogenous transformations.

For the Aizu-Bandaisan dance, normally the former objective function is used, except for the period of put hands together that the latter function is used. The weight w is set by trial and error during performing simulation.

The reason that the second objective function is not always used in case that robot has different size of each body part from human is that using objective function that optimizes both joint angles and hand position may cause joint angles to shift little from the value it should be due to redundancy in robot arm. If the robot has the same size as human, which is often not the case, using the first or second objective function should not have much difference.

Note that in our work, the first and last 3 control points of B-spline curve is set to be equal in the objective function, so that from equations (3.10) and (3.11), the velocity and acceleration at the begin and the end of curve would be 0. Of course zero velocity is needed at such points for practical motion. For zero acceleration, the force at the needed to drive actuator should be lower. In other words, zero acceleration results in controller friendly trajectory.

Until this step, this work is similar to most of the previous works that perform optimization based on data representation technique. Next the advantages of this work are introduced, that the constraints for optimization can be derived from B-spline function.

5.2 Constraint

In the history of robotic path planning, there are many constraints that are suggested to be considered. In order to realize the applicable trajectory, often more than 1 constraint should be used. Some authors say that important ones are angle/velocity/acceleration limits, others says angle/velocity/force limits, or even only angle/force limits, and so on. Which actually depends on system itself.

In any cases B-spline function can be exploited as constraints, which here describes the usage of 4 important ones of them: angle, velocity, acceleration, and force limit. After that the composition of angle/velocity/acceleration limits are considered for general cases, and the composition of angle/velocity/force limits are considered in the case of our humanoid robot that uses harmonic actuator.

Angle limit

From the characteristic of B-spline that its amplitude will not be higher than magnitude of control point, angle limit can be done directly by applying bound constraint to the magnitude of control points.

In other words, use angle ranges of each joint as the bound constraint for optimization.

Velocity limit

Velocity limit is serious one because actuator often has back emf that increases proportionally with velocity. Too much of back emf can damage the actuator. So it is strongly suggested to limit velocity in our case of using humanoid robot HRP2 with harmonic actuator. Now the method of exploiting cubic B-spline as velocity constraint is shown below.

As mentioned in chapter 3, since each knot period of cubic B-spline is functioned with 4 control points, the equation of the velocity of cubic B-spline at each knot period that has the length of 1 is:

$$\dot{S}(t) = \frac{1}{6}\{(-3t^2 + 6t - 3)p_0 + (9t^2 - 12t)p_1 + (-9t^2 + 6t + 3)p_2 + (3t^2)p_3\} \quad (5.3)$$

The velocity at the beginning and at the middle of knot period can be calculated by inserting 0 and 0.5 to the above equation, respectively.

$$\dot{S}(0) = \frac{p_2 - p_0}{2} \quad (5.4)$$

$$\dot{S}(0.5) = \frac{p_3 + 5p_2 - 5p_1 - p_0}{8} \quad (5.5)$$

Then generalize it in the case that the period length is different from 1:

$$\dot{S}(begin) = \frac{p_2 - p_0}{2T} \quad (5.6)$$

$$\dot{S}(middle) = \frac{p_3 + 5p_2 - 5p_1 - p_0}{8T} \quad (5.7)$$

while T is the length of knot period.

The length of knot period is selected to be able to represent the low frequency motion of the trajectory to be optimized.

With these generalized equations, velocity at the beginning and the middle of each period can be limited by adjusting the value p_2 as this algorithm:

```
Vbegin=(p2-p0)/2/T;
if (Vbegin>Vub)
   temp1=p0+2*T*Vub;
elseif (Vbegin<Vlb)
   temp1=p0+2*T*Vlb;
end;
```

```
Vmiddle=(p3+5p2-5p1-p0)/8/T;
if (Vmiddle>Vub)
   temp2=(-p3+5*p1+p0+8*T*Vub)/5;
elseif (Vmiddle<Vlb)
   temp2=(-p3+5*p1+p0+8*T*Vlb)/5;
end;

if (Vbegin>Vub and Vmiddle>Vub)
   p2=min(temp1,temp2);
elseif (Vbegin<Vlb and Vmiddle<Vlb)
   p2=max(temp1,temp2);
elseif (Vbegin>Vub or Vbegin<Vlb)
   p2=temp1;
elseif (Vmiddle>Vub or Vmiddle<Vlb)
   p2=temp2;
end;
```

This algorithm can be described as follow. If the velocities are larger the limit at both positions, voting is done to select the value of p_2 that controls both velocities to be smaller than limit.

Furthermore, is there any case that velocity at the beginning of any period is above upper limit while velocity at the middle is lower than lower limit, or vice versa. The answer is yes. By assuming that the acceleration curve be like:

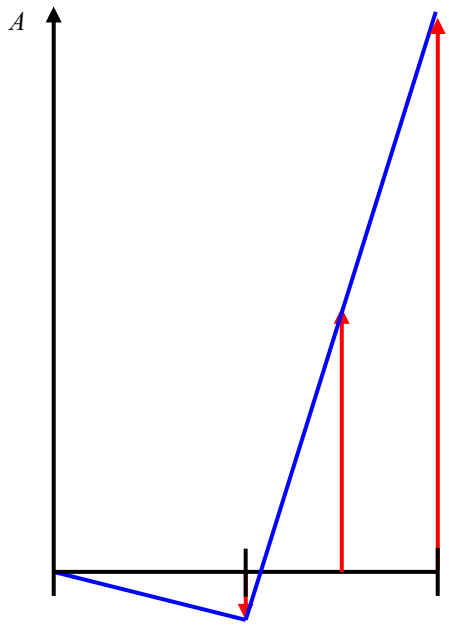

Figure 5.1 Illustration 1 of acceleration for calculating velocity

In such case, by integrating the acceleration curve with velocity equals to 0 at the initial of curve, velocity at the beginning of 2nd period would be negative, while velocities at the middle and the end of such period would be positive.

It can be assumed further that both value are larger than limit. Since the conflict happens as the sign of such 2 points different, the algorithm will choose to solve velocity limit (assumed to be $\pm T/2$) of at the first period by increasing p_2:

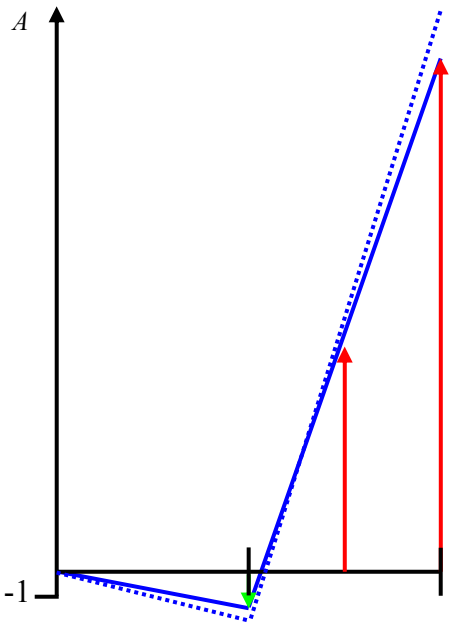

Figure 5.2 Illustration 2 of acceleration for calculating velocity

It can be seen that the end point of acceleration curve also changes, because changing a control point of cubic B-spline affects maximum 4 periods of knot vector. From equation of acceleration, increasing acceleration at the beginning of any period by the value k will decrease acceleration at the end of such period by the value of $2k$:

$$\ddot{S}_n(begin) = \frac{(p_2 + k) - 2p_1 + p_0}{T^2}$$
$$\ddot{S}_{n+1}(begin) = \frac{p_3 - 2(p_2 + k) + p_1}{T^2} \quad (5.8)$$

However, it can be seen that the acceleration curve does not change much, that velocity in the middle of 2nd period is still larger than limit. The problem might is there, what should be done.

The answer is nothing needed to be done since at the next iteration velocity at the beginning of 3rd period would be limited, thus velocity at the middle of 2nd period is solved automatically.

To verify this assumption, as you can see that if the velocity limit is $\pm T/2$, acceleration at the beginning of 3rd period would be 3 so that velocity would be limited as desired:

$$V_{begin\ of\ 3rd\ period} = \frac{1}{2}T(-1) + \frac{1}{2}\frac{T}{4}(-1) + \frac{1}{2}\frac{3T}{4}(3) = \frac{T}{2} \tag{5.9}$$

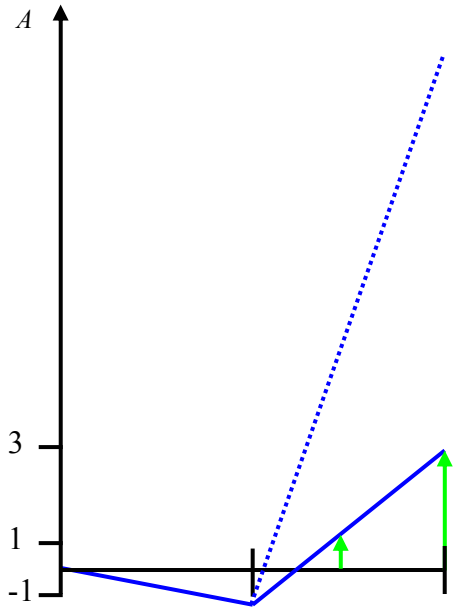

Figure 5.3 Illustration 3 of acceleration for calculating velocity

Then the velocity at the middle of 2nd period is also limited.

$$V_{middle\ of\ 2nd\ period} = \frac{1}{2}T(-1) + \frac{1}{2}\frac{T}{4}(-1) + \frac{1}{2}\frac{T}{4}(1) = \frac{-T}{2} \tag{5.10}$$

Actually from illustration 3 of acceleration here, another question may arise in the mind of reader, what about the velocity at the zero acceleration crossing at the quarter of 2nd period.

$$V_{quarter\ of\ 2nd\ period} = \frac{1}{2}T(-1) + \frac{1}{2}\frac{T}{4}(-1) = \frac{-5T}{8} \tag{5.11}$$

which is lower than limit value $-T/2$ here.

This phenomenon is not viewed as problem in this work. Obviously since the acceleration is straight line, the peak of velocity often not occur at the beginning or the middle of period, but be as above case. In that example, the ratio between peak value and limit value is:

$$(\frac{-5T}{8})/(\frac{-T}{2}) = 1.25 \tag{5.12}$$

Actually this is the worst ratio that can happen. To verify that the 1.25 ratio is the worst case, let's see another example from illustration 4 of acceleration. Assume that the velocities at the beginning of 3rd and 4th period are larger than limit value:

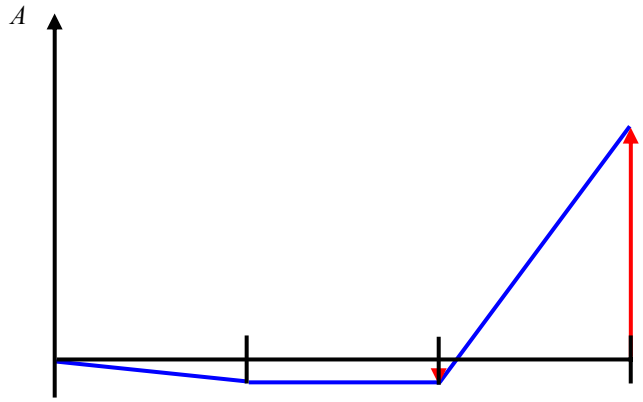

Figure 5.4 Illustration 4 of acceleration for calculating velocity

that after solving for velocity at the beginning of 3rd period:

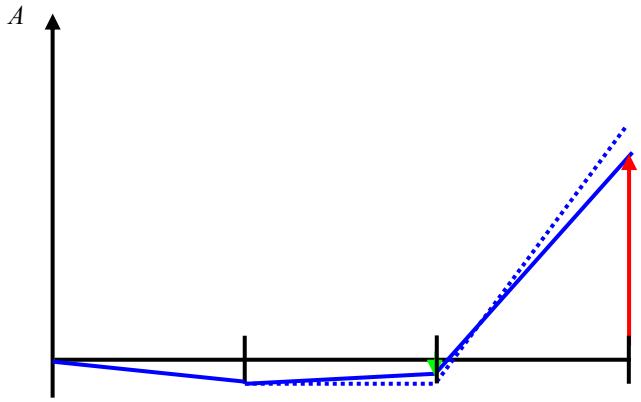

Figure 5.5 Illustration 5 of acceleration for calculating velocity

and solve for the velocity at the beginning of 4th period:

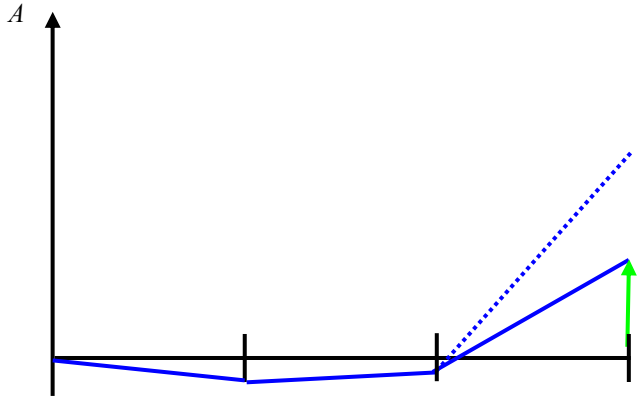

Figure 5.6 Illustration 6 of acceleration for calculating velocity

At the zero acceleration crossing, the peak negative velocity in the 3rd period is very close to the limited velocity at the beginning of that period, such that the ratio between them is obviously below 1.25.

So it is clear that from this illustration, the peak velocity at zero acceleration crossing is limited by the value of 1.25 times of used limit. Hence the actual velocity limit, as verified from the worst case above, would be 1.25 times of used limit.

Another method would be updating the value of p_2 until the peak velocity yield limit. However since such updating makes the position of peak value changes, so it requires checking and updating iteratively.

The worse thing about iterative method is changing p_2 of next period may affects the peak value of previous period in an unwanted manner. As can be seen from illustration 6 of acceleration that altering p_2 of the last period results in the larger peak velocity. If one thinks that this problem is unacceptable both mathematically and especially practically that the peak value might be greater than limit, it is recommended to used the previous 1.25 limit criterion instead.

Acceleration limit

Since the second derivative of cubic B-spline is straight line, acceleration limit is quite simpler than that of velocity limit. As mentioned in chapter 3, since each knot period of cubic B-spline is functioned with 4 control points,the equation of the acceleration of cubic B-spline at each knot period that has the length of 1 is:

$$\ddot{S}(t) = \frac{1}{6}\{(-6t+6)p_0 + (18t-12)p_1 + (-18t+6)p_2 + (6t)p_3\} \qquad (5.13)$$

The acceleration at the beginning of each period can be calculated from inserting 0 to the above equation, respectively.

$$\ddot{S}(0) = p_2 - 2p_1 + p_0 \qquad (5.14)$$

$$\ddot{S}(0.5) = \frac{p_3 - p_2 - p_1 + p_0}{2} \qquad (5.15)$$

Then generalize it in the case that the period length is different from 1:

$$\ddot{S}(begin) = \frac{p_2 - 2p_1 + p_0}{T^2} \qquad (5.16)$$

$$\ddot{S}(middle) = \frac{p_3 - p_2 - p_1 + p_0}{2T^2} \qquad (5.17)$$

while T is the length of knot period.

Now, since the 2nd derivative curve of each period is a straight line, only the acceleration at the beginning of each knot period is enough for limiting such value by using simple algorithm like:

```
Abegin=(p2-2*p1+p0)/T^2;
if (Abegin>Aub)
   P2=2*p1-p0+T^2*Aub;
elseif (Abegin<Alb)
   P2=2*p1-p0+T^2*Alb;;
end;
```

Force limit

Manipulator's actuator has many limits due to both the actuator itself and manipulator structure. Three main limits are angle velocity and force. For force, it can be calculated from inverse dynamics equation:

$$F_i = \sum_j M_{ij}(Q)\ddot{Q}_j + Ia\ddot{Q}_i + \sum_j \sum_k C_{ijk}(Q)\dot{Q}_j\dot{Q}_k + G_i(Q) \qquad (5.18)$$

where F is applied force at joint i, M_{ii} is effective inertia at joint i, M_{ij} is coupling inertia between joint i and j, Ia is actuator's inertia at joint i, C_{ijj} is centripetal forces at joint i due to velocity at joint j, C_{ijk} is Coriolis forces at joint i due to velocities at joint j and k, and G_i is gravitational loading at joint i.

Noted that from now on in this subsection, Einstein summation is used, and all coefficients are the function of Q except Ia.

In joint space, the path can be parameterized by the function of single scalar $f(r)$ According to Shin [25], applied force can be limited by:

$$F_i^{\min} \le M_{ij} f_j' \ddot{r} + (M_{ij} f_j'' + C_{ijk} f_j' f_k')\dot{r}^2 + G_i \le F_i^{\max} \qquad (5.19)$$

where f', f'' indicate 1st and 2nd derivative of $f(r)$ by r.

In our case, the path is parameterized by control point of B-spline that actually s is time, and also rotor inertias are provided. The above equation is adjusted to limit value by inserting velocity and acceleration in the form of B-spline. Since the acceleration is straight line so its peak values situate at the beginning of each knot period and the acceleration multiplies to the term M_{ij} that has a lot of influence to dynamic equation, so limiting force only at the beginning of knot period is practically sufficient. Note that since the dynamic equation's parameters are nonlinear to Q, so they are viewed as constants that are updated iteratively.

From (5.6) and (5.16):

$$F_i = M_{ij}(\frac{p_2 - 2p_1 + p_0}{T^2})_j + Ia(\frac{p_2 - 2p_1 + p_0}{T^2})_i + C_{ijk}(\frac{p_2 - p_0}{2T})_j(\frac{p_2 - p_0}{2T})_k + G_i \quad (5.20)$$

Then separate the function to the terms that consist and not consist of p_2.

$$\begin{aligned} F_i = {} & M_{ij}(\frac{p_2 - p_0}{T^2})_j + M_{ij}(\frac{-2p_1 + 2p_0}{T^2})_j + \\ & Ia(\frac{p_2 - p_0}{T^2})_i + Ia(\frac{-2p_1 + 2p_0}{T^2})_i + \\ & C_{ijk}(\frac{p_2 - p_0}{2T})_j(\frac{p_2 - p_0}{2T})_k + G_i \end{aligned} \quad (5.21)$$

Finally rearrange the term and apply the limit:

$$\begin{aligned} F_i^{\min} \le {} & [\frac{2M_{ij}}{T} + C_{ijk}(\frac{p_2 - p_0}{2T})_k](\frac{p_2 - p_0}{2T})_j + \frac{2Ia}{T}(\frac{p_2 - p_0}{2T})_i + \\ & M_{ij}(\frac{-2p_1 + 2p_0}{T^2})_j + Ia(\frac{-2p_1 + 2p_0}{T^2})_i + G_i \le F_i^{\max} \end{aligned} \quad (5.22)$$

which is a set of n nonlinear equations and n variables (p_2), that is different from [25] since they find $\ddot{r}$ that satisfy $\max_i LB_i \le \ddot{r} \le \min_i UB_i$ which is not necessary in the case that trajectory is the function of time, such as B-spline.

The above equation can be solved by iteratively changing p_2 of joint i alone. Since the force equation is nonlinear, sometimes the initial condition is not good so that changing p_2 may be locked at local minima. Hence, although the force is to be minimized until it meets limit, it is allowed that updated force be larger than previous iteration's force for a limit number of allowance. At each knot period:

```
%Calculate initial value
for i=1:joint
   K1(i)=(p2(i)-p0(i))/(2*T);
   K2(i)=(-2*p1(i)+2*p0(i))/T^2;
   Q(i)=(p2(i)+4*p1(i)+p0(i))/6;
end;
```

```
%Start force limit
for i=1:joint

   F=0; temp=0; myBreak=0;

   while (temp==0 | F<Flb(i) | F>Fub(i))

      K4=0;

      [M,C,G]=calculate_dynamic_parameter(Q,i);

      for j=1:joint

         K3=0;

         for k=1:joint
            K3=K3+C(i,j,k)*K1(k);
         end;

         K4=K4+(2*M(i,j)/T+K3)*K1(j)+M(i,j)*K2(j);

      end;

      F=K4+2*Ia(i)/T*K1(i)+Ia(i)*K2(i)+G(i);

      %Check whether present position stuck in local minima or not
      if (abs(F)>=abs(Fold))

            myBreak=myBreak+1;

            if myBreak>=maxAllowable
               break;
            end;

      end;

      Fold=F;

      %If force exceeds limit, change p2(i) for a small value, del
      K=2*M(i,i)/T+2*Ia(i)/T+
        sum_over_k(C(i,i,k)*(p2(i)-p0(i))k/2T)

      if (F<Flb(i))
         if (K>0)
            p2(i)=p2(i)+del;
         elseif (K<0)
            p2(i)=p2(i)-del;
         end;
```

```
        elseif (K>Fub(i))
           if (K>0)
              p2(i)=p2(i)-del;
           elseif (K<0)
              p2(i)=p2(i)+del;
           end;

        end;

        %Update parameters that related to p2(i)
        K1(i)=(p2(i)-p0(i))/(2*T);
        Q(i)=(p2(i)+4*p1(i)+p0(i))/6;
        temp=temp+1;

     end;

  end;
```

Actually, from the above algorithm, the force limit itself has problem of sensitivity. Though the dynamic force equation of each joint consists of multiple input angles, for simplicity, limiting each joint's force is done by adjusting the angle of that joint alone. It is found that the sensitivity of changing each joint's parameters namely angle velocity and acceleration toward result force is sometimes low, depending on the posture of manipulator. This means it might require a lot of change to trajectory to meet with the force limit. Now, this problem is lessened by imposing allowable change of joint's parameters from initial value.

Composition of angle/velocity/acceleration limits

In many works the composition of angle/velocity/acceleration is used without dynamic force limit. Actually the force limit is more important than acceleration limit since the force limit represents the dynamic physical limit of manipulator directly, that varies from posture to posture of robot. Also most of the actuators' specifications provide limits in term of velocity and force, with recommended voltage and current. Then with the understanding of manipulator structure such as gear ratio and friction, force limit for such manipulator can be calculated.

However since the dynamic force equation is difficult to solve, and it requires a lot of data not only from the actuator itself but also from the manipulator structure, sometimes using acceleration limit instead can be a nice choice.

How does acceleration limit represent force limit. As can be seen from dynamic force equation, the input to such equation is angle, velocity, and acceleration of manipulator's joints. So limiting acceleration generally also results in lower force.

First thing to mention about this composition of limit is that angle limit can be done directly by limiting the magnitude of control points as stated in previously in this chapter. Next the velocity and acceleration limits are considered. Based on both limits that are derived previously, the voting algorithm is as:

```
Vbegin=(p2-p0)/2/T;

if (Vbegin>Vub), temp1=p0+2*T*Vub;
elseif (Vbegin<Vlb), temp1=p0+2*T*Vlb;
end;

Abegin=(p2-2*p1+p0)/T^2;

if (Abegin>Aub), temp2=2*p1-p0+T^2*Aub;
elseif (Abegin<Alb), temp2=2*p1-p0+T^2*Alb;;
end;

Vmiddle=(p3+5p2-5p1-p0)/8/T;

if (Vmiddle>Vub), temp3=(-p3+5*p1+p0+8*T*Vub)/5;
elseif (Vmiddle<Vlb), temp3=(-p3+5*p1+p0+8*T*Vlb)/5;
end;

if (Vbegin>Vub and Abegin>Aub and Vmiddle>Vub)
   p2=min(temp1,temp2,temp3);
elseif (Vbegin<Vlb and Abegin<Alb and Vmiddle<Vlb)
   p2=max(temp1,temp2,temp3);

elseif (Vbegin>Vub and Abegin>Aub)
   p2=min(temp1,temp2);
elseif (Vbegin<Vlb and Abegin<Alb)
   p2=max(temp1,temp2);

elseif (Vbegin>Vub and Vmiddle>Vub)
   p2=min(temp1,temp3);
elseif (Vbegin<Vlb and Vmiddle<Vlb)
   p2=max(temp1,temp3);

elseif (Abegin>Aub and Vmiddle>Vub)
   p2=min(temp2,temp3);
elseif (Abegin<Alb and Vmiddle<Vlb)
   p2=max(temp2,temp3);
```

```
elseif (Vbegin>Vub or Vbegin<Vlb)
   p2=temp1;
elseif (Abegin>Aub or Abegin<Alb)
   p2=temp2;
elseif (Vmiddle>Vub or Vmiddle<Vlb)
   p2=temp3;

end;
```

Here, the priority of important is given to velocity at the beginning of the period first. Then it is the acceleration. The velocity at the middle of period is considered lowest priority since even though it is not solved in the present period, it would be solved in the next period, as discussed in the velocity limit section.

The voting between velocities at the beginning and the middle of period is already discussed previously in this chapter. Though conflict of velocities between the beginning and the middle can happen as seen in illustration 1 of acceleration for calculating velocity can be solved automatically as discussed, in the case that acceleration is also taking part, there might be flaw in voting algorithm, which we have not found yet. Actually this algorithm is not used, since the information about the actuator and manipulator is provided, the method that is used in this work includes force limit. Which is explained next.

Composition of angle/velocity/force limits

Again, angle is limited by the bounded constraint. As mentioned before, generally the velocity limit is more important than acceleration limit, and the velocity alone is limited effectively so that many motions can be precisely represented by the robot.

However for the abrupt motion the dynamic force limit is also needed, as you can see that although the velocity that is over limit may not occur in the case that robot lift up package slowly however if it is too heavy the force of actuator would obviously not enough. In this case the constraint should consist of both force and velocity limits. While doing the experiment in real robot, it is strongly suggested to use the input motion that the velocity is limited. In this case, the constraint routine is set to use both limits, and the result will meet velocity limit while sometimes fail for force one.

The method that meets both limits simultaneously is yet to be found, since the force is iteratively reduced before explicitly limit the velocity. However the force after performing velocity limit does not change much so it is not such severe problem because in practical the instantaneous force can be higher than limited value. In other

words, the instantaneous current needed to drive actuator can be higher than rated ampere.

Although the optimization routine converges by a certain value based on minimizing an objective function and a set of constraints, however convergence does not means that the error between original angle and optimized angle are low enough. The obvious case is that if the control point density is too low for an original angle, no matter how many the iterations pass, the convergence is not satisfied.

So, let's consider the topics about refinement the precision of B-spline curve, by detecting the problems of optimized curve and solving them in the hierarchical B-spline structure.

5.3 Error detector

For the case of using hierarchical structure to refine the precision, traditional error detector is normally used.

After optimization is finished, in order to detect the error, the optimized B-spline curve is compared to original angle. Then if the error of any part larger than a certain value, that part of curve would be marked as having problem.

$$e(n) = | \theta(n) - q(n,[p]_{opt}) | > thErr \tag{5.23}$$

where *thErr* is the threshold for error, given by user.

5.4 Density detector

The density problems comprise of two issues. First, a set of B-spline has limitation in the sense that it cannot represent a curve that swings more frequently than some certain threshold. Second, in contrary, too dense B-spline can result in over optimized B-spline curve, which swings more frequently than original curve.

It is found that detecting the swing pattern of each joint angle and assigning appropriate density of B-spline, would help solving the density problems.

In the below figure, B-spline density is too low around the time of 1 second.

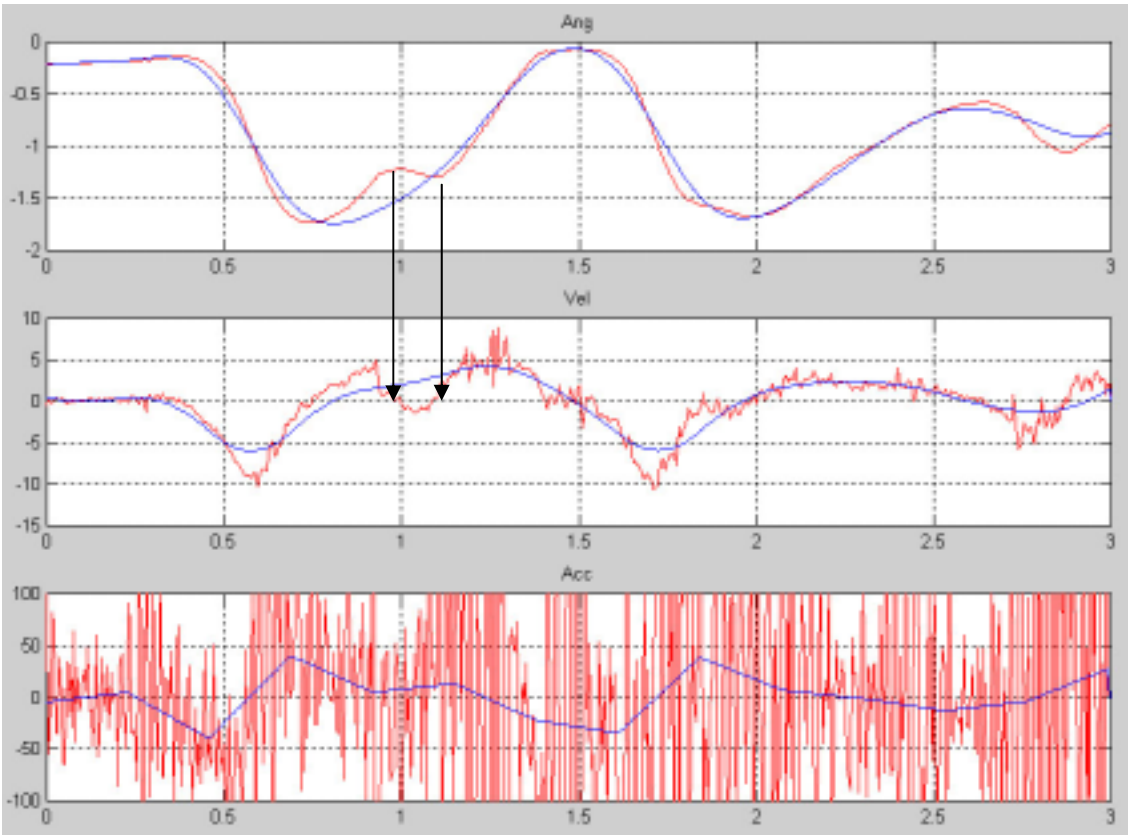

Figure 5.6 Density of B-spline is too low, red is original curve, blue is B-spline

From this example, it can be seen that the distance between 2 peak values in the angle indicates the swing frequency. So, in order to detect the peak value in angle, velocity's zero crossing should be detect as above figure. However the original curve is derived from digital motion capture system, hence its velocity has spike noises that are needed to be suppressed. So frequency based filtering is used here.

Since the sampling rate is 200 Hz, the frequency range of motion signal is 100 Hz. The pass band is set to pass the high frequency motion of human which here is selected to be 6 Hz, and the stop band is 10 Hz. These two values should not be too close to each other since the order of filter would increase. Longer filter has longer delay, so the problem that the end of curve is not presence in the filtered signal can occur.

At first, infinite impulse response filter or IIR filter is selected since it has sharp cut off with low order. It is designed by using Chebychev type 1 that error in pass band can be controlled. However IIR's phase distortion is not linear as shown in

figure below. The filter with nonlinear phase has the problem that delay corresponding to each frequency is not equal, from the equation:

$$delay(\omega) = -\frac{d}{d\omega}\{\arg[H(e^{j\omega})]\} \tag{5.24}$$

The delay that is not constant is not good for positioning problem, such as the zero crossing detection.

So finite impulse response filter or FIR filter is used instead since it has linear phase as in phase figure below. Though normally FIR requires higher order than IIR, in zero crossing detection problem cut off needs not to be sharp. So low order filter that has small delay can be selected.

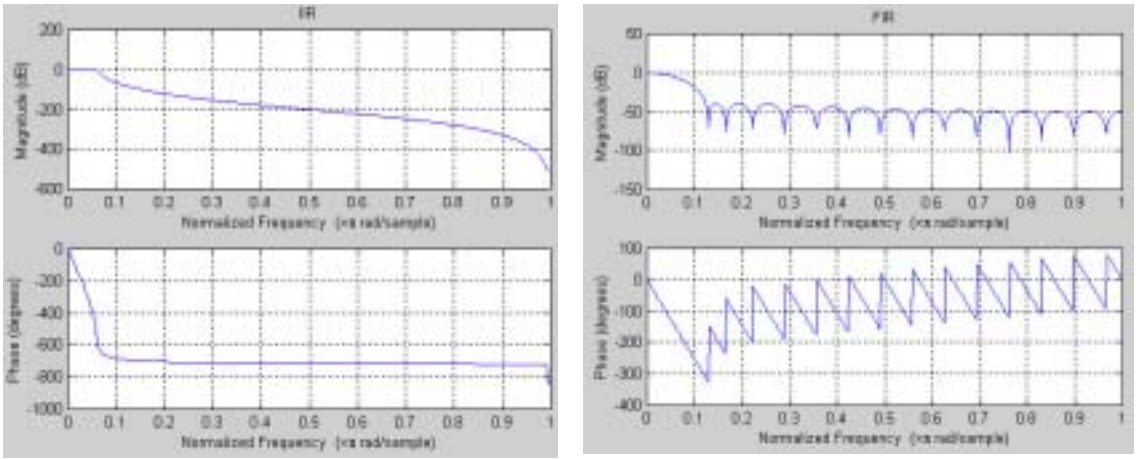

Figure 5.7 Magnitude and phase of IIR v.s. FIR

Then the results from these 2 filters with the delay of FIR even lower that that of IIR can be seen in the next figure. Although there is not so much different in the efficiency of these 2 filters, however the group delay of FIR that is calculated directly from phase is constant, while this is not the case for IIR.

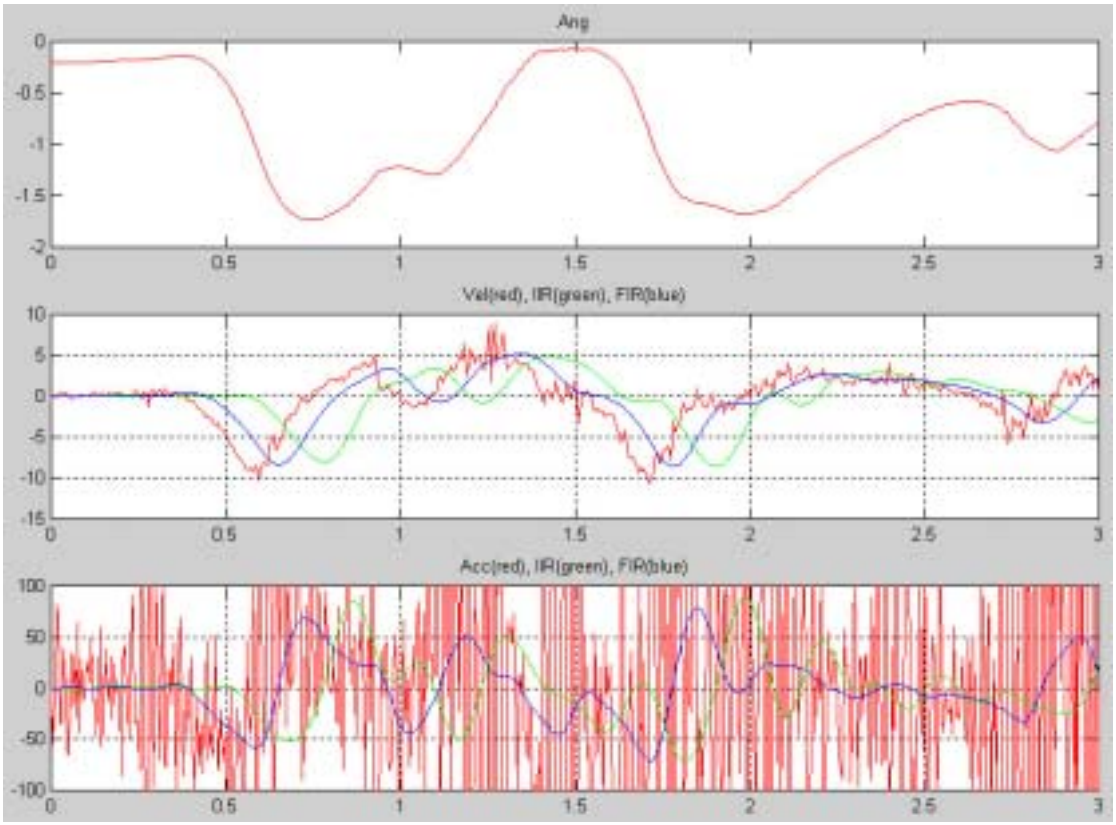

Figure 5.8 Filtered results of IIR v.s. FIR

Be careful that sampling rate of signal should be as high as possible to reduce delay. For example, an FIR with delay of 14 samplings, if sampling rate is 200 fps the delay is 0.07s, however if sampling rate is 50 fps the delay is 0.28 s, which is not suit for zero crossing detection problem. So we use the frame rate of 200 fps without down-sampling signal before filter.

B-spline has characteristic that knot period T that can represent data with peak-peak range as small as T itself, as it can be seen from below figure:

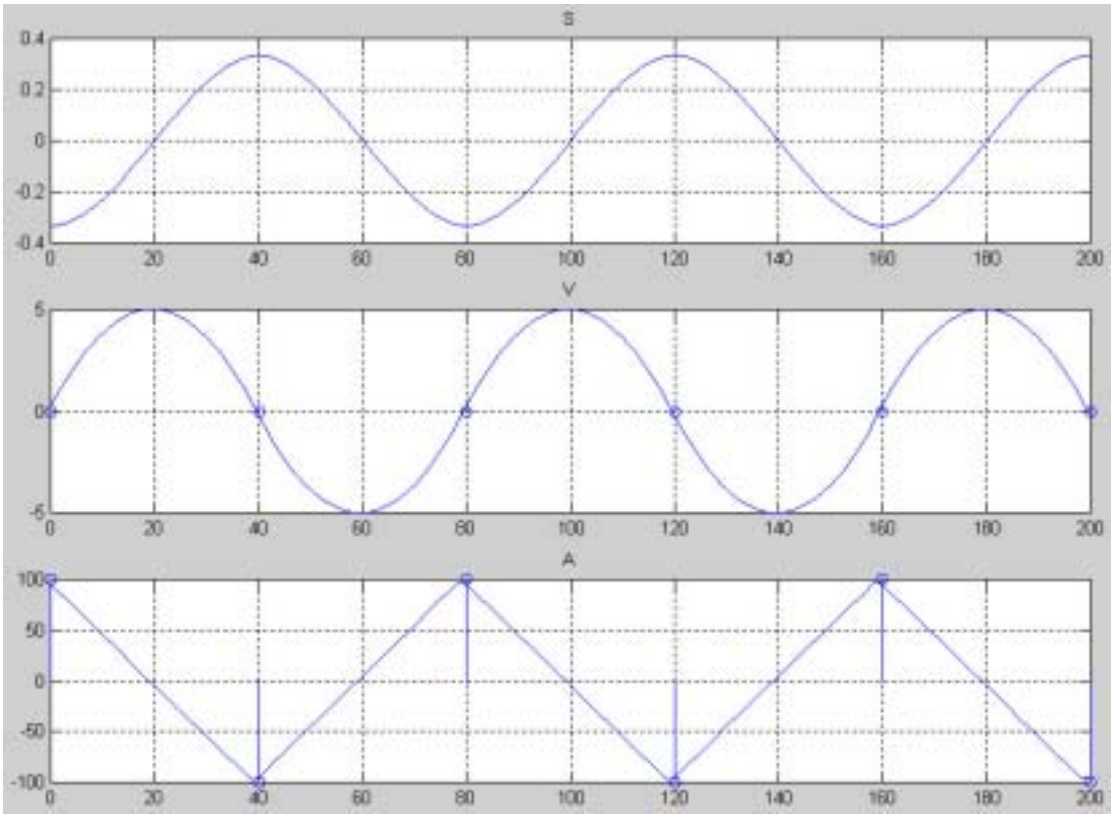

Figure 5.9 Relation between peak-peak period and knot period

From this example, the knot period, clearly seen from velocity or acceleration graph, is 40 frames. While the peak-peak of signal or velocity's zero crossing period is also 40 frames, or 0.2 s in the case of sampling rate of 200 fps.

So if 2 zero crossing points of velocity are detected with in the range below the present knot period, here is 0.2 s, that part of curve cannot be represented by the present B-spline. That it will be fed into hierarchical structure. Furthermore if the range is below 0.2 s and equal or above 0.1 s, it will be assigned to 1st hierarchy. If the range is below 0.1 s, it will be assigned to 2nd hierarchy.

Though the hierarchical B-spline optimization is not explained, let's see the better output from it in the below figure, compared to the one in the beginning of this section.

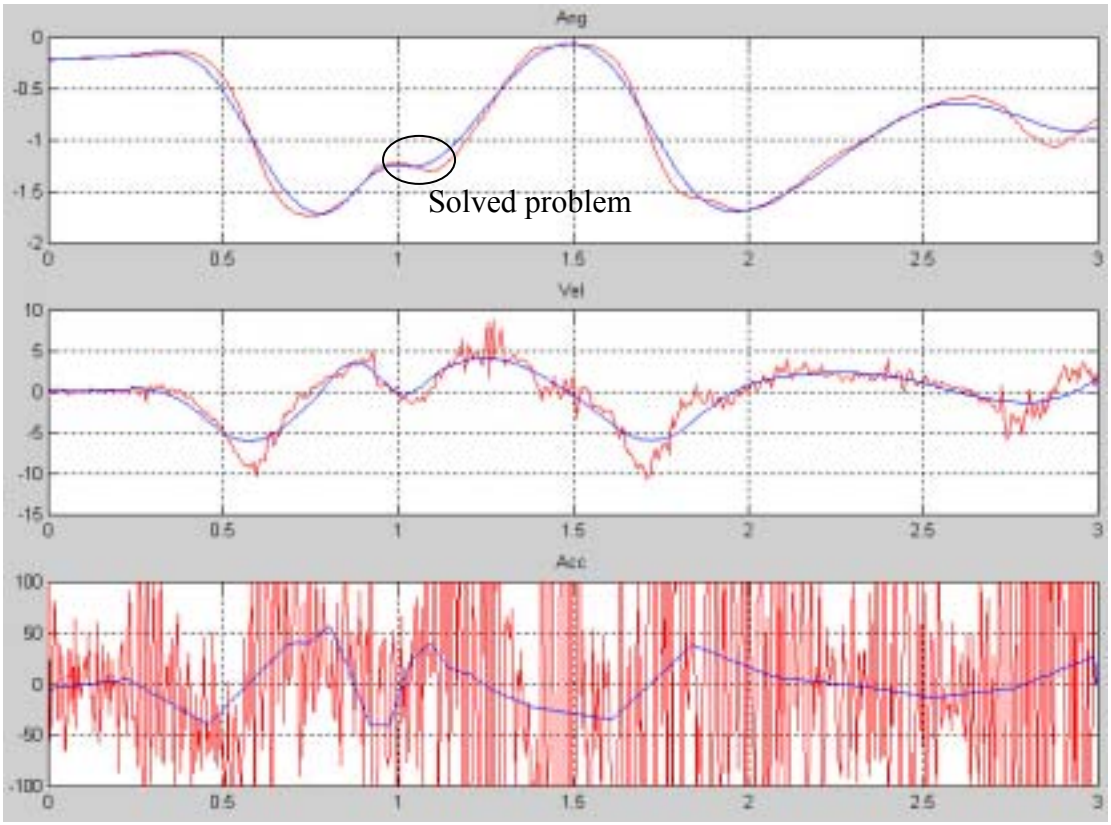

Figure 5.10 Density of B-spline is appropriate, red is original curve, blue is B-spline

5.5 Hierarchical B-spline

From the previous 2 sections we are now ready to refine the precision of trajectory. That if the error of any part of B-spline curve is larger than threshold or if the input joint angle swings more frequently than a certain threshold that cannot be represent by the current B-spline, that set of B-spline coefficients will be fed to hierarchical B-spline decomposition, to get new B-spline that has greater number of control points.

$$[hp] = (hp_0, hp_1, ..., hp_{hcp}) = Hie([p]_{opt}) = Hie(p_0, p_1, ..., p_{cp}) \qquad (5.25)$$

where $[hp]$ is a set of $hcp + 1$ hierarchical control points hp and $hcp > cp$, Hie denotes the decomposition function.

From Cohen [3], Chui [1], or Forsey [6], the decomposition function can be easily implemented based on the knowledge that a basis of B-spline can be represented by 5 half-width bases, with the coefficients:

$$h = \frac{1}{8}\{1,4,6,4,1\} \tag{5.26}$$

Then perform decomposition by the equation:

$$hp_i = \sum_{j=0}^{i} h_{i-2j}\, p_j \tag{5.27}$$

, and the number of hp_i is:

$$\begin{aligned} 2 * length(in) + length(h) - 2 &= 2 * (cp + 1) + 5 - 2 \\ &= 2 * cp + 5 \end{aligned} \tag{5.28}$$

, the factor 2* is used since the length of output is double of input, and the factor –2 is used the same way with well-known convolution that output sequence's length is the sum of inputs length minus 2.

Furthermore since periodic cubic B-spline is used, so the first 3 and last 3 control points, that each basic has different form from bell shape basis, will not be used, that the actual length is:

$$2 * cp + 5 - 6 = 2 * cp - 1 \tag{5.29}$$

, compared to $cp + 1$ input.

Then this new set of control points, only at the period of joint angle that has problem, will be optimized again to find a new optimum set of B-spline.

$$[HP]_{opt} = \underset{[HP]}{\arg\min} \sum_{n=1}^{N} f(\Theta(n) - Q(n,[HP])) \tag{5.30}$$

where $[HP]$ is the sets of hierarchical control points for all considered angles.

5.6 Hierarchical B-spline's constraint

Since the hierarchical B-spline will be optimized only at problem period, the process of changing B-spline's coefficient in constraint functions must be adapted. Consider when a control point changes, the consequence is 4 knot periods are affected. This poses no problem in the case of using just B-spline, because the any constraint will be checked simultaneously until the end of curve. However for hierarchical structure, the control points to be optimized are not all of them, and that the problem like below example could occur.

Assume the number of B-spline's control point is 6. After decomposition, from the former section, the number will be 9, and that there are 6 knot periods. The points to be re-optimized are the 5^{th} one. Let's use the acceleration limit to understand the change. Assume further that acceleration limit is ± 100. After re-optimization, the magnitude of 5^{th} control point is altered to be higher. From the acceleration limited described, the acceleration at the beginning of 3^{rd} knot period is checked, however no limitation is performed at 4^{th} and 5^{th} periods. Consequently the acceleration at the beginning of 4^{th} period is larger than limit in figure a) below.

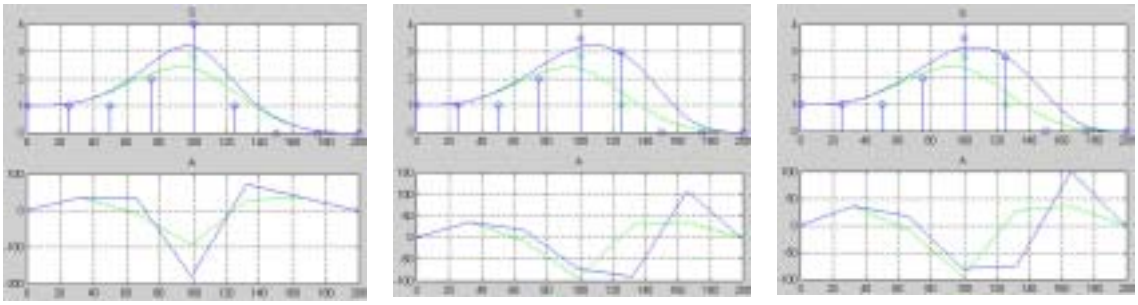

Figure 5.11 Optimize: a) 1 control point b) 2 control points c) with backward

So if we want that point to be checked, increase the re-optimized control point to be 2. This time the acceleration at the beginning of both 3^{rd} and 4^{th} periods is limited. However the problem happens again at 6^{th} period as in figure 5.11 b).

In order to solve this problem, you might already saw that the acceleration constraint should be performed not only forward but also backward, results in figure 5.11 c).

That is to say, the problem of unchecked knot period occurs because the number of affected knot periods is higher than the number of re-optimized control points by the factor of 3. This problem is solved from performing limit checking both forward and backward. This kind of limiting is used in our work for most of described constraints.

It is important to note here that performing limiting backward can affect prior (left hand side) knot period in an unwanted manner that the value, such as acceleration, at prior knot period may become higher than limit. In order to cope with this possibility, the number of control points to be re-optimized is set to be at least 6 points in our work. So that in the backward run, the flaw after adjusting 6^{th} and 5^{th} control points would be resolved by adjusting 4^{th} and 3^{rd} control points, and the flaw after adjusting 4^{th} and 3^{rd} control points would be resolved by adjusting 2^{nd} and 1^{st} control points. Though 6 prove to be effective in our work, user may choose higher number of control point such as 8 or 10. Actually it can be checked if problem occurs or not, if there is, assign the higher number of re-optimized control points.

Chapter 6

Result

The Aizu-Bandaisan dance is used as test subject for this work. This dance is very complicated that original motions exceed many physical limits of robot, so it is not easily performed by humanoid robot with existing methods. If our algorithm can deal with such dance, it would be benefit to other kind of motions also.

The results shown in this chapter are aimed for discussing the efficiency of objective function, constraints, and hierarchical structure. They comprised of both experimental and simulation results. It will be shown that angle, velocity, and acceleration limits are can be explicitly controlled. While force limit can be almost 100 percents implicitly controlled by iterative search program, however it is allowed to have the instantaneous force larger than limit value since the sensitivity issue as shown in the previous chapter.

Note that only 4 out of 7 joints in each arm, 3 in shoulder and 1 in elbow, that affect the position and orientation very much are passed through optimization program, while joints 5 and 6 that do not move so much and are not relate to end-effector position are sufficiently performed only angle and velocity limits by [22]'s method, joint 7 in hand is always constant since it is not used in this dance.

6.1 Result of using objective function

The objective function that preserves joint angles as in (5.1) of course proves to be efficient. However since the markers that are attached to the body of human via elastic suit may not balance, slip during capturing, or even occluded that the position sequences are derived from interpolation, that the angles after inverse kinematics can have large error. In these cases, error can be reduced by adjusting the end-effector of

robot to the hand position (not error prone orientation, in this case) of human as in (5.2). It is used in this work by comparing wrist positions between human and robot, at the period that synchronization of hand positions are matter such as one shown below.

In the below motion, the markers of left and right arms are not balance, as can be seen that length of human's arm created from markers' positions are not equal. After performing inverse kinematics, the error in the positions of robot's end-effectors is apparent during the motion of put hand together. So around such motion, (5.2) is used with *w* equals to 5.

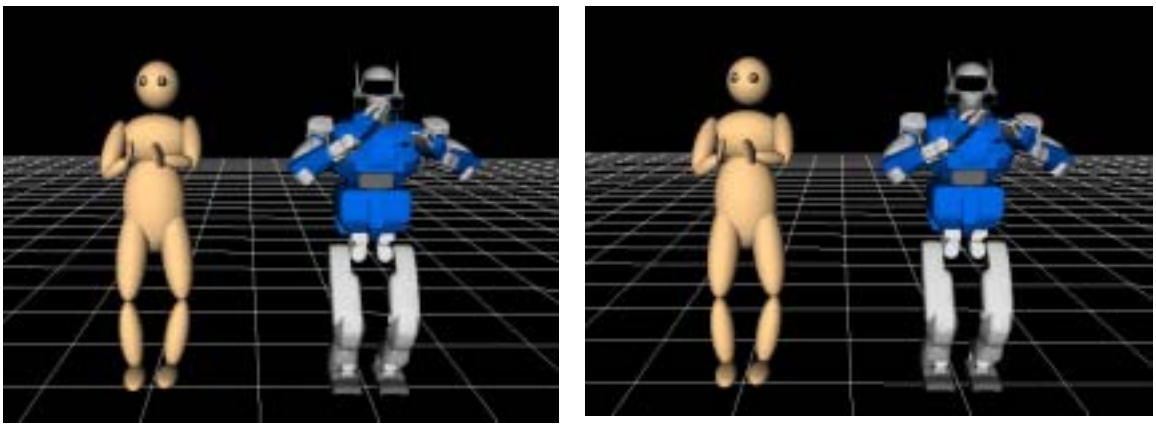

Figure 6.1 Comparing arms of human figure generated from markers's postions v.s. Left) Result after inverse kinematics Right) Result after optimization

6.2 Result of using constraints

The trajectories shown afterward are mostly constrained by the physical limits received from AIST. For the limits that are not acquired, acceleration limit are assumed and simulated because it is not actually used in real robot, while force and instantaneous force limits are calculated form experimental data.

Also there are discussions about using hierarchical structure inside, which point out that if the robot abruptly moves a force limit is important to reduce the dynamic force, although velocity limit is still not exceeded.

Note that in this section red line means original data after inverse kinematics, blue line means optimized data, and dotted magneta line means actual data in robot.

Angle limit

The angles limits of HRP2's right arm and left arm are [-3.1416 to 1.0472, -1.6581 to 0.1745, -1.6057 to 1.6057, -2.3911 to 0.0349] and [3.1416 to 1.0472, -0.1745 to 1.6581, -1.6057 to 1.6057, -2.3911 to 0.0349] radians, respectively. Usually it is recommended to set the limit range smaller than actual range to support the dynamic movement. In our case the buffer range of 5 degrees or 0.0873 radians is used.

By applying the bounded constraints to the magnitude of B-spline, here is the result in joint angle 3 of left hand that is actually tested in HRP2.

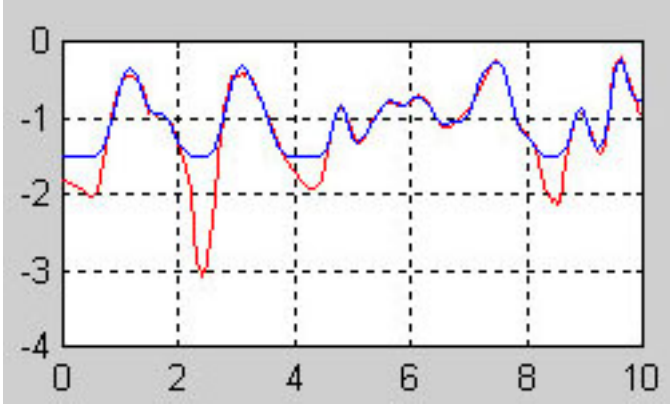

Figure 6.2 Angle limit

Velocity limit

The velocitys limits of HRP2's right arm are [4.17044, 2.4192, 4.15699, 2.23776] radians per second.

By applying algorithm of velocity limit, here are the results of velocity limit of 70, 80, 90 percents respectively using just only mere B-spline so that the precision is not so good. Anyway, these results can actually be used in robot. Note that peak velocity can be 1.25 times of limit value and angle limit is also used.

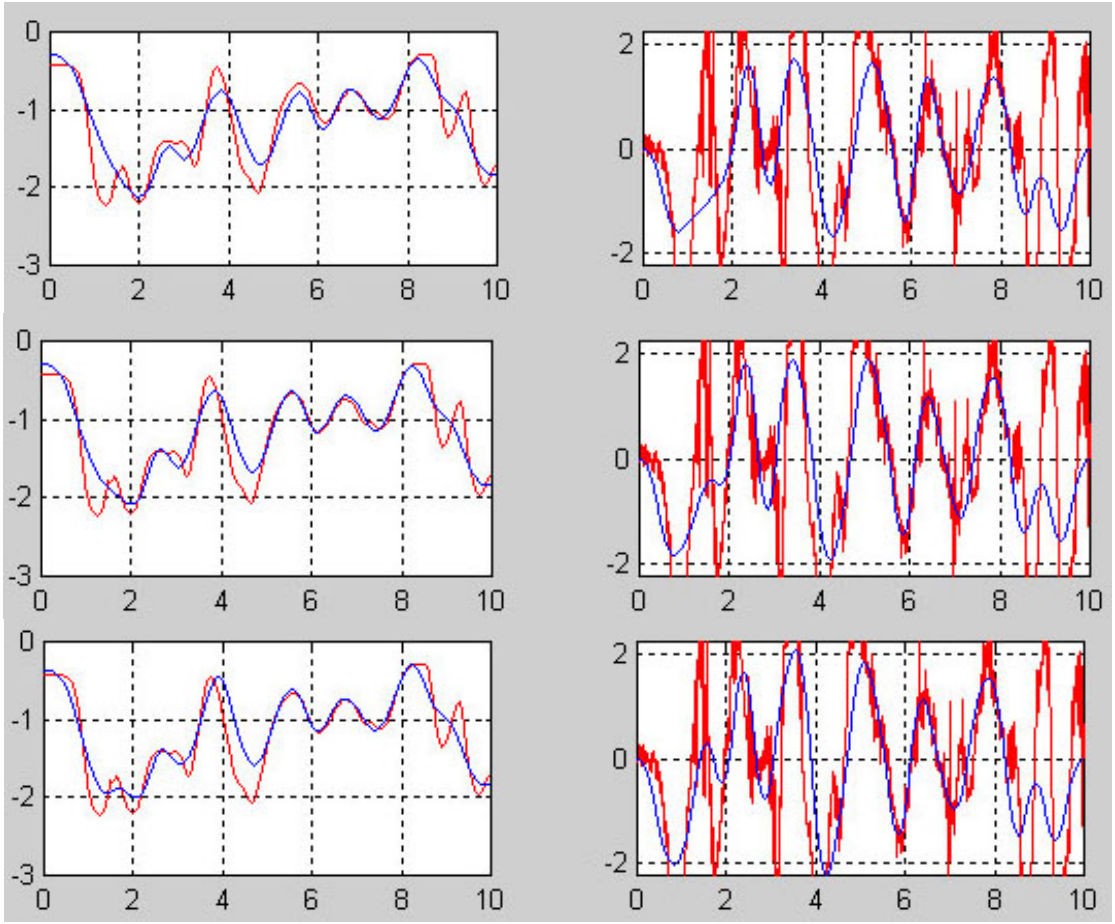

Figure 6.3 Velocity limits of 0.7, 0.8, 0.9, respectively, based on B-spline
Left column shows angle and right column shows velocity

It can be seen that the precision is not so good so the hierarchical structure is applied. However hierarchical curves abruptly change like original angle, among velocity limit of 0.7 0.8 and 0.9 only the data with limit of 0.7 can be used in real robot. Since the robot system reports that problem occurs at joints 3 of right hand, let's look at experimental data compared to optimized data for angle velocity and force of joint 3 with velocity limit of 0.8. The problem is not from angle or velocity that are lower than limits for both optimized and actual trajectories.

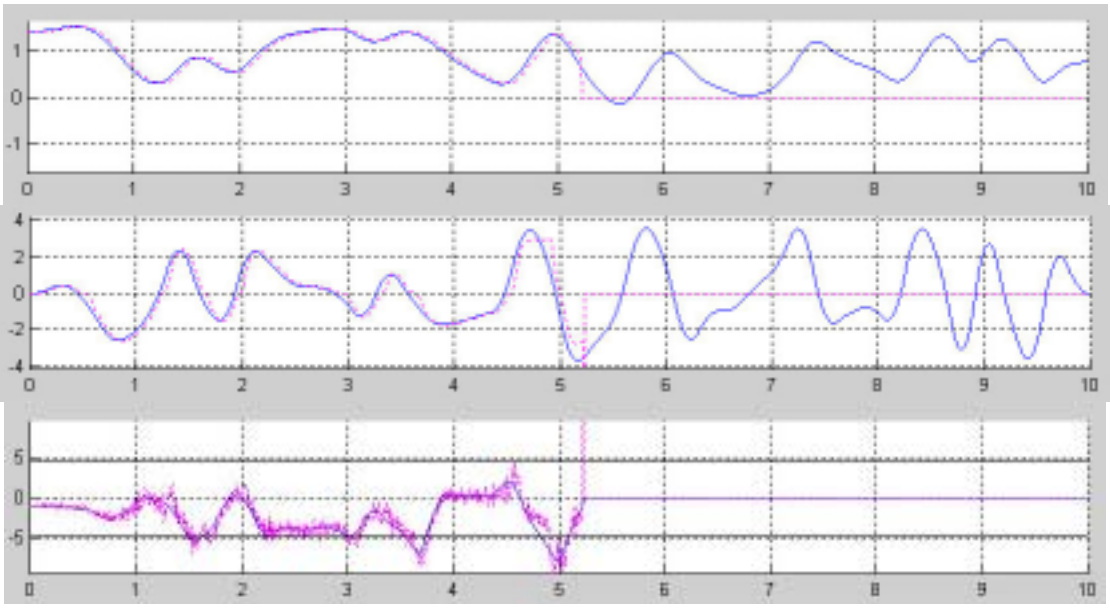

Figure 6.4 Velocity limit of 0.8 based on Hierarchical B-spline
Angle velocity and force, respectively

In the above figure, angle and velocity is plotted in the limit range, while force is plot in the range of 2 times of limit, with black line as limit. It can be seen that before the robot stop, instantaneous force is very large compared to limit, especially the actual data that force exceeds 2 times of limit. Hence the force limit is important

Acceleration limit

Since the acceleration limit is not provided, so it is checked by only simulation. Here is the result of using angle limit from actual limit, while velocity limit is set to be 5, and acceleration limit is set to be 50.

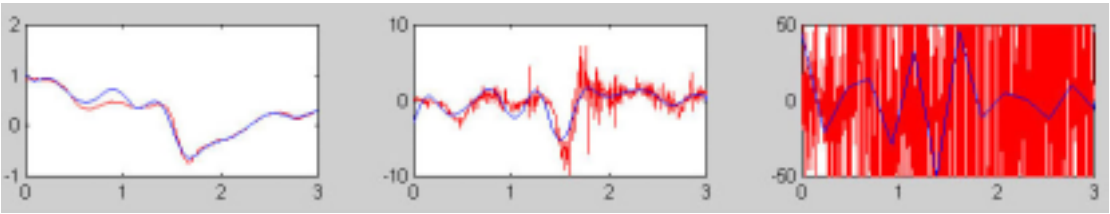

Figure 6.5 Composition of angle/velocity/acceleration limits
Angle velocity and acceleration, respectively

Force limit

The simulation results of optimization with no limit at all, force limit 100 percents, and force limit 150 percents, respectively, are shown below.

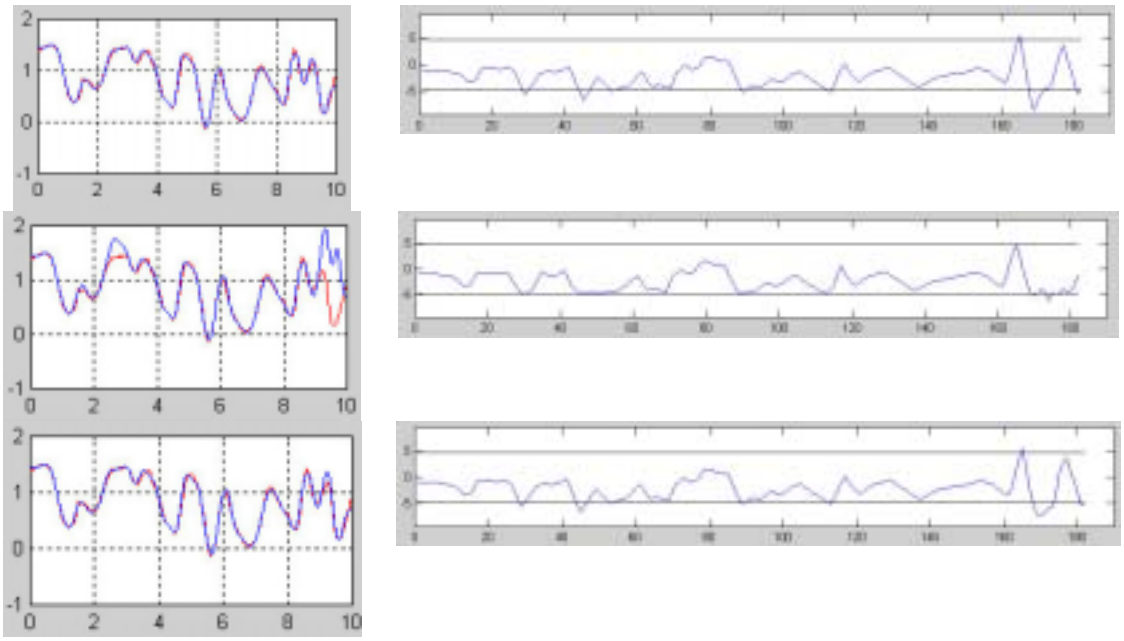

Figure 6.6 Force limits of infinity, 1.0, 1.5, respectively
Left column shows angle and right column shows force

It can be seen that though the force can be limited almost perfect, since the sensitivity of changing a trajectory toward force is not good, so it is better to allow the instantaneous force to happen.

Composition of angle/velocity/force limits

At the present time after, the compositions of these limits are not yet tested, since the robot is sent to have repairing. So here are the simulation results of using 80 percents of angle limit and velocity limit, while force limit is set to be infinity and 150 percents. Note that the simulation results can also be compared to last section since the angle here does not exceed the limit, so the differences are only from velocity and force limits.

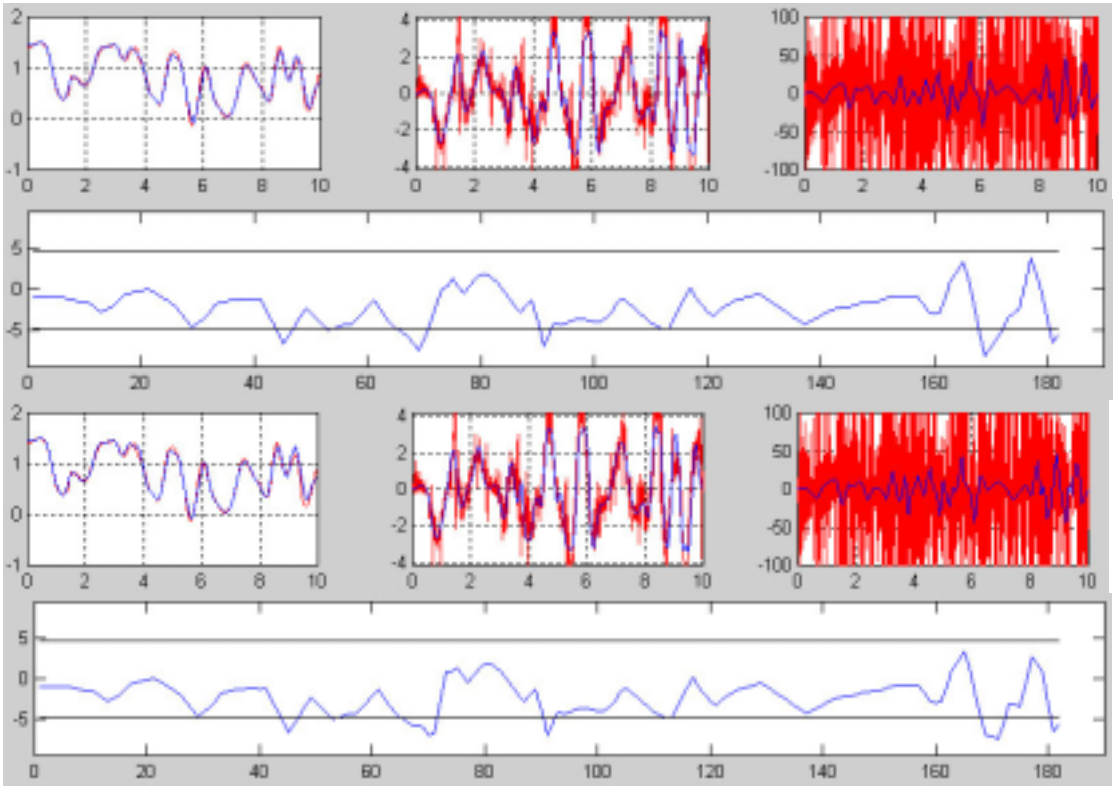

Figure 6.7 Angle limit, velocity limit of 0.8, force limits of infinity and 1.5 Shown in order of angle, velocity, acceleration, and force.

It can be seen that angle and velocity are well limited to value stated, original velocity range is about 100 percents of limit, while force limit of 1.5 results in similar trajectory with lower instantaneous force that should be benefit that problem likes figure 6.4 would not occurs.

Also comparing to previous figure in the case force limit equals to infinity, using velocity limit could result in higher force, as you can see in the middle of force trajectory. The method that limits both velocity and force simultaneously and effectively is yet to be found.

Chapter 7

Conclusion

Our research group focuses on motion control by using robot off-line to preserve Japanese traditional dances, which are considered as intangible cultural assets. However, some of them are disappearing because of a lack of successors. Using humanoid robot also popularizes these national assets. At present Aizu-Bandaisan, which is the traditional dance from Fukushima prefecture in Japan, is used as test motion. This dance is very complicated that original motions exceed many physical limits of robot, so it is not easily performed by humanoid robot with existing methods. If our algorithm can deal with such dance, it would be benefit to other kind of motions also.

From original human motions, trajectory optimization of humanoid robot with physical limits as constraints is presented. The optimization's objective function is subjected to preserve salient characteristic of the original motion, while constraints are used to transform the motion to the capabilities of the humanoid robot.

Unlike other previous works of humanoid robot trajectory generation, using constraints to limit physical characteristic values would ensure that limits are met, better than just reducing them by objective function.

Though recently, using B-spline wavelets is shown to be the method that has some advantages over hierarchical B-spline in the field of motion generation, of both humanoid robot and animation. For humanoid robot, physical limits are very important, and B-spline coefficients can be exploited in constraints for optimization better. It is explained how to use such coefficients in angle, velocity, acceleration, and dynamic force constraints.

Frankly speaking, our work still has limitation that, although velocity or acceleration or both constraints can be limited explicitly by B-spline structure,

dynamic force equation is generally nonlinear so it is solved by iterative method. This poses two problems.

First, the force limit itself has problem of sensitivity. In our work, though the dynamic force equation of each joint consists of multiple input joint angles, for simplicity, limiting each joint's force is done by adjusting the angle of that joint alone. It is found that the sensitivity of changing each joint's parameters namely angle velocity or acceleration toward result force is sometimes low, depending on the posture of manipulator. This means it might require a lot of change to trajectory to meet with the force limit. At present, this problem is solved by allowing instantaneous force that is larger than limit to happen, which actually can be done in practical to some certain value derived from experiment.

Second, for humanoid robot that uses harmonic actuator, the velocity limit is more important than acceleration limit, and the velocity is limited effectively that many motions can be precisely represented by the robot. However for the abrupt motion, the dynamic force limit is also needed. In this case the constraint must consists of both force and velocity limit. When the experiments are tested in real robot, it is allowed only to use the input motion that the velocity is limited. In this case, the constraint routine is set to use both limits, and the result will meet velocity limit while sometimes fail for force one. We still cannot find the method that meets both limits simultaneously since the force is iteratively reduced. However this is not severe problem because in practical the instantaneous force can be higher than limited value.

Even in the present of these problems, this work should work better than just minimizing velocity or acceleration as previous optimization based methods, since the angle velocity and acceleration can be limited effectively, and force can be limited for sure if the need be. Even if the force-velocity limit is used, the force is reduced effectively.

7.1 Future work

The force limit should be improved, base on the knowledge that the dynamic force equation of each joint consists of multiple input joint angles, so limiting each joint's force could be done by adjusting related joints' angles such as reducing force in the elbow joint by adjusting shoulder joint.

Furthermore the dynamic force equation at present is not the best representation of actual manipulator since the friction are unknown, so the equation that also consists of friction terms should be exploited as force constraint for optimization.

Appendix

A1. Jacobian based Inverse Kinematics

As mentioned in chapter 2, certain kind of inverse kinematics can be exploited as trajectory optimization, it will be shown that jacobian based inverse kinematics can be assigned objective function to reduce some physical constraints of manipulator. The recent works that specifically explores the algorithm for humanoid robot or high DOFs manipulator are Tevatia [28] and D'Souza [4]. However presented here is only the basic theory of jacobian based inverse kinematics.

In order to derive the formula, let's use an example of 2 dimensions as shown below.

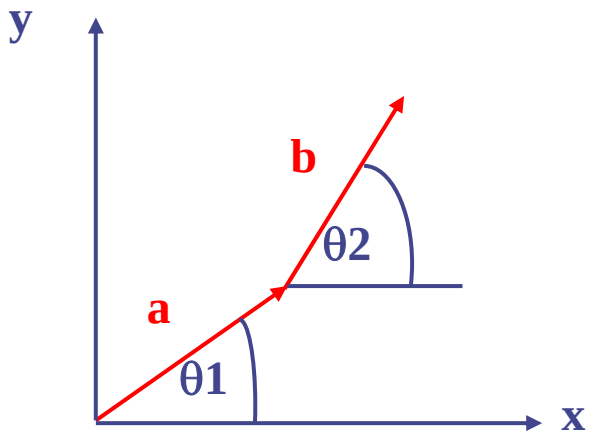

Figure A1.1 Vector in 2 dimensional space

From the above figure, the forward kinematics function:

$$x = a\cos\theta 1 + b\cos\theta 2$$
$$y = a\sin\theta 1 + b\sin\theta 2 \qquad \text{(A1.1)}$$

Differentiate:

$$\begin{aligned} \dot{x} &= -a\dot{\theta}1\sin\theta 1 - b\dot{\theta}2\sin\theta 2 \\ \dot{y} &= a\dot{\theta}1\cos\theta 1 + b\dot{\theta}2\cos\theta 1 \end{aligned} \tag{A1.2}$$

Rearrange in matrix form:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} -a\sin\theta 1 & -b\sin\theta 2 \\ a\cos\theta 1 & b\cos\theta 1 \end{bmatrix} \begin{bmatrix} \dot{\theta}1 \\ \dot{\theta}2 \end{bmatrix} \tag{A1.3}$$

Or $\dot{S} = J\dot{Q}$, which can be solved for $\dot{Q}$ by taking the inverse of J if it is square i.e. m=n, and non-singular. For a redundant manipulator n is greater than m, e.g., human arm or leg that has 7 joints with 6 parameters of end-effector, x-y-z-rot_x-rot_y-rot_z, which necessitates the use of additional constraints to obtain a unique inverse.

There are mainly 2 methods for solving the above problem, pseudo-inverse method, and extended Jacobian method. The former is quite expensive for highly redundant system, the latter is more suitable with highly redundant system, though generally time consuming, [28]. Since we are dealing with robot arm that is not highly redundant, pseudo-inverse method will be used.

Pseudo-inverse methods are a common choice to invert in face of redundancies, e.g. by using the Moore-Penrose inverse:

$$\dot{Q} = J^{\Phi}\dot{S} = J^{T}(JJ^{T})^{-1}\dot{S} \tag{A1.4}$$

Which the characteristic of it can be described in equations:

$$\begin{aligned} \mathrm{J}\mathrm{J}^{\Phi} &= \mathrm{J}\mathrm{J}^{\mathrm{T}}(\mathrm{J}\mathrm{J}^{\mathrm{T}})^{-1} = \mathrm{I} \\ \mathrm{J}^{\Phi}\mathrm{J} &= \mathrm{J}^{\mathrm{T}}(\mathrm{J}\mathrm{J}^{\mathrm{T}})^{-1}\mathrm{J} \neq \mathrm{I} \end{aligned} \tag{A1.5}$$

Liegeois [15] suggested a more general form of optimization with pseudo-inverses by minimizing an explicit objective function G in the null space of J:

$$\dot{Q} = J^{\Phi}\dot{S} - (I - J^{\Phi}J)\frac{\partial G}{\partial Q} \tag{A1.6}$$

where $(I - J^{\Phi} J)$ is null space of J :

$$J\dot{Q} = JJ^{\Phi}\dot{S} - (J - JJ^{\Phi}J)\frac{\partial G}{\partial Q} = \dot{S} \tag{A1.7}$$

If partial different term of objective function is positive, Q' is decreased, that G is decreased. And if it is negative, Q' is increased, that G is decreased.

However there are drawbacks of inverse kinematics style optimization. First, it needs redundancy to do optimization. If there is no such redundant joint, it will be reduced to below equation that has no meaning for optimization.

$$\dot{Q} = J^{-1}\dot{S} - (I - J^{-1}J)\frac{\partial G}{\partial Q} = J^{-1}\dot{S} \tag{A1.8}$$

Second, although it can reduce the objective function locally, however this reduced result is not mathematically guaranteed to be optimal result.

Third, this method suit with the case that only end-effector position (and orientation) is provided. In our work, position and orientation of most of joint angles are sufficiently provided from markers attached on human, so optimization of each angle's trajectory is more suitable. Also the method described in this work does not have such above 2 problems.

A2. Gimbal lock

Since the joint of human robot is not flexible as that of human, the problem of Gimbal lock can happen. Gimbal lock can be easily explained as the position where joint angle of robot cannot move to allow manipulator to go to target position as shown in figure below.

Actually whether the Gimbal lock problems will occur in the trajectory optimization or not depends on many conditions, such as the objective function, the data representation technique, and the optimization method. In other words, although the angle is in the lock position, the problem may not happen if that present condition allows manipulator to move. Let's consider each condition.

First, if the objective function for such motion as in figure below is controlling hand position, it is likely that robot will not move. However if the objective function both controls hand position and reduces the power of robot, obviously the hand will move lower to reduce actuator power.

Second, with the objective function that controls hand position, if the optimization is done at each time frame, the robot will not move because moving any angle only result in the same or farer distance from the target. However if the data representation technique is used, the problem turns to be curve optimization. Since there is distance between control points in such technique, the trajectory can allow the hand to jump to better position.

Third, with the objective function that controls hand position based on data representation technique, if local optimization is used, the hand position is not guaranteed to move to target position since the result trajectory may be local minimum. However if the global optimization is used so that many minima are searched, hand can move to target position, which is global minimum.

We suggest that before choosing the optimization method, between local and global, the objective function should be considered. Because using global optimization generally consumes a lot more time than local one.

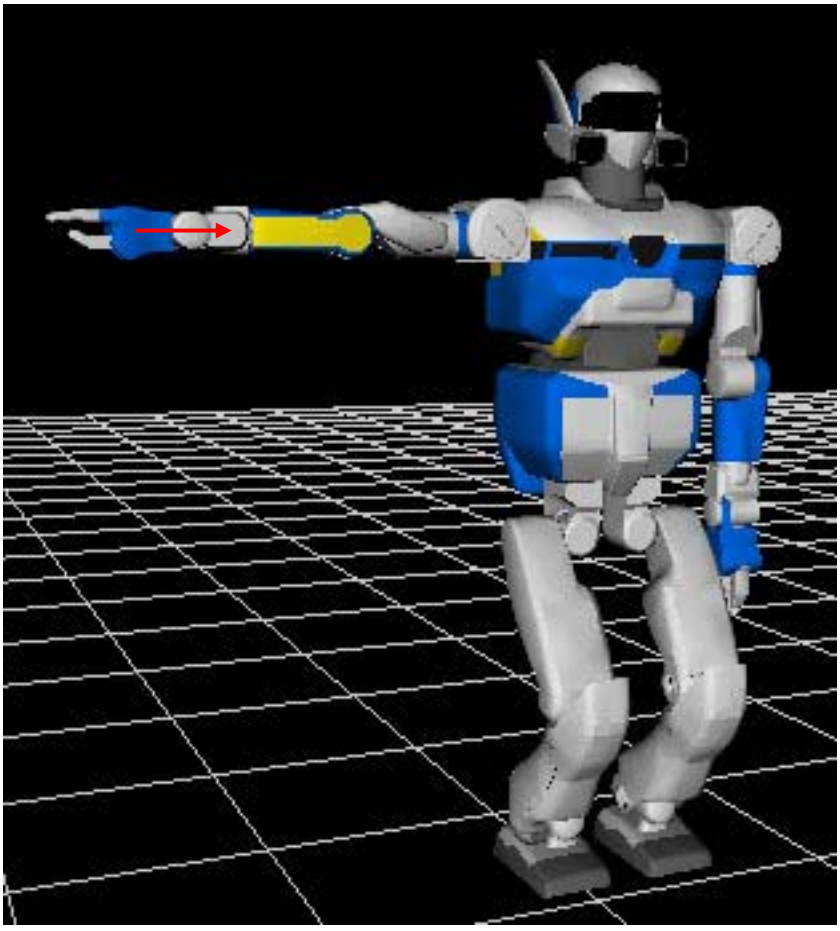

Figure A2.1 Gimbal lock, hand cannot move to target position

References

[1] Chui, C.K., and Quak, E., An Introduction to Wavelets, *Wavelet Analysis and its Application*, Academic Press, 1992, vol. 1.

[2] Cohen, E., Daubechies, I., and Feauveau, J.C., "Biorthogonal bases of compactly supported wavelets", *Communication on Pure and Applied Mathematics 45*, 1992, pp. 485–560.

[3] Cohen, E., Lyche, T., and Riesenfeld. R., "Discrete B-splines and subdivision techniques in computer-aided geometric design and computer graphics", *Computer Graphic and Image Processing 14*, vol. 2, pp. 87–111, October 1980.

[4] D'Souza, A., Vijayakumar, S., and Schaal, S., "Learning inverse kinematics", *IEEE International Conference on Intelligent Robots and Systems,* 2001, pp. 298-303.

[5] Endo, K., Maeno, T., and Kitano, H., "Co-evolution of morphology and walking pattern of biped humanoid robot using evolutionary computation – consideration of characteristics of the servomotors", *IEEE International Conference on Intelligent Robots and Systems,* 2002.

[6] Forsey, D., and Bartels. R., "Hierarchical B-spline refinement", *Computer Graphic 22*, pp. 205–212, August 1988.

[7] Gleicher, M., "Retargetting motion to new characters", *ACM SIGGRAPH*, 1998, pp. 33–42.

[8] Gotler, S.J., "Hierarchical and variational geometric modeling with wavelets", *ACM Symposium on Interactive 3D Graphics*, 1995, pp. 35-42.

[9] Hattori, M., Kitada, S., Tadokoro, S., and Takamori, T., "The motion description in computer to make and edit body movement data", *PNC Annual Conference and Joints Meetings 2001 PNC/ECAI/IPSJ-SIGCH/EBTI*, 2001.

[10] Huyer, W., and Neumaier, A., Multilevel coordinate search, Kluwer Academic Publishers, 1998, http://www.mat.univie.ac.at/~neum/software/ls/.

[11] Kajita, S., "A realtime pattern generator for biped walking", *IEEE International Conference on Robotics and Automation,* 2002.

[12] Kaneko, K., Kanehiro, F., Kajita, S., Yokoyama, K., Akachi, K., Kawasaki, T., Ota, S., and Isozumi, T., "Design of prototype humanoid robotics platform for hrp", *IEEE International Conference on Intelligent Robots and Systems*, 2002.

[13] Korb, W., and Torch, I., "Data reduction for manipulator path planning", *Robotica*, Cambridge Unicersity Press, 2003,vol. 21, pp. 605-614.

[14] Lee, J., and Shin, S.Y., "A hierarchical approach to interactive motion editing for human-like figures" *ACM SIGGRAPH*, 1999, pp. 39-48.

[15] Liegeois., A., "Automatic supervisory control of the configuration and behavior of multibody mechanisms", *IEEE Transactions on Systems, Man, and Cybernetics*, 1977, 7(12), pp. 868–871.

[16] Luenberger, D.G., *Optimization by vector space method*, John Wiley & Sons, 1969.

[17] Lyche. T., and Morken, K., "Knot removal for parametric B-spline curves and surfaces", *Computer Aided Geometric Design*, vol. 4, pp. 217-230, 1987.

[18] Nagasaki, T., Kajita, S., Yokoi, K., Kaneko, K., Tanie, K., "Running pattern generation for a humanoid robot", *Annual Conference of the Robotics Society of Japan 19*, 2000.

[19] Nakamura, M., and Hachimura, K., Labanotation and new technology: application of hypermedia to choreography and dance education, 2000

[20] Nakaoka, S., Nakazawa, A., Yokoi, K., Hirukawa, H., and Ikeuchi, K., "Generating whole body motions for a biped humanoid robot from captured human dances", *IEEE International Conference on Robotics and Automation,* 2003, pp. 3905-3910.

[21] Nishiwaki, K., Kagami, S., Kunitoshi, Y., Inaba, M., and Inoue, H., "Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired zmp", *IEEE International Conference on Intelligent Robots and Systems,* 2002.

[22] Pollard, N.S., Hodgins, J.K., Riley, M.J., and Atkeson, C.G., "Adapting human motion for the control of a humanoid robot", *IEEE International Conference on Robotics and Automation,* 2002.

[23] Rose, C., Guenter, B., Bodenheimer, B., and Cohen, M.F., "Efficient generation of motion transitions using spacetime constraints", *ACM SIGGRAPH*, 1996.

[24] Safonova, A., Pollard, N.S., and Hodgins, J.K., "Optimizing human motion for the control of a humanoid robot", *2nd International Symposium on Adaptive Motion of Animals and Machines*, 2003.

[25] Shin, K.G., and McKay, N.D., "Minimum-time control of robotic manipulators with geometric path constraints", *IEEE Transactions on Automatic Control,* 1985.

[26] Soga, A., "Web3d dance composer: a web-based ballet performance simulation system", *International Symposium on Electronic Art 11*, 2002, pp. 16-19.
[27] Tak, S., Song, O.Y., and Ko, H-S, "Motion balance filtering", *EUROGRAPHICS*, vol. 19, no. 3, 2000.
[28] **T**evatia, G., and Schaal, S., "Inverse kinematics for humanoid robot", *IEEE International Conference on Robotics and Automation,* 2000.
[29] Ude, A., Atkeson C.G., and Riley M., "Planning of joint trajectories for humanoid robots using B-spline wavelets", *IEEE International Conference on Robotics and Automation*, 2000.
[30] Whitney, D.E., "Resolved motion rate control of manipulators and human prostheses", *IEEE Transactions on Man-Machine Systems*, 1969, 10(2): pp. 47-59.
[31] William II, R.L., "Local performance optimization for a class of redundant eight-degree-of-freedom manipulators", *NASA Technical Paper*, 3417.
[32] Yamane, K., and Nakamura, Y., "Efficient parallel dynamics computation of human figures", *IEEE International Conference on Robotics and Automation,* 2002.
[33] Yokoi, K., Kanehiro, F., Kaneko, K., Fujuwara, K., Kajita, S., and Hirukawa, H., "A Honda humanoid robot controlled by aist software", *IEEE-RAS International Conference on Humanoid Robots*, 2001.
[34] Yukawa, T., Kaiga, T., Nagase, K., and Tamamoto, H., "Human motion description system using buyo-fu" (in Japanese), *Information Processing Society of Japan*, 2000.