We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

1

Automatic Trajectory Generation using Redundancy Resolution Scheme Based on Virtual Mechanism

Bojan Nemec and Leon Žlajpah
Robotics Laboratory, Jožef Stefan Institute, Jamova 39, 1001 Ljubljana Slovenia

1. Introduction

In recent times, while markets are reaching their saturation limits and customers are becoming more demanding, a paradigm shift has been taking place from mass production to customized mass production. The concept of customization focuses on satisfying a customer's unique needs with the help of new technologies. Typically, the products are similar but they differ in some parameters which make manual teaching and manual preparation of the manufacturing programs not acceptable. The customized mass production requires that all production phases are prepared in advance during the design phase of the specific product. This requires that standard production procedures are modified and prepared for each specific product. It is also required that the adaptation is done automatically without any human intervention. In modern production systems, CAD models of the product are used to generate specific machining programs. In the case of industrial robots, automatically generated programs have to consider various limitations, such as joint limits, wrist singularity and possible collisions of the robot with the environment. Although the off-line programming enables detection of such situations during the program preparation, it does not solve the basic goal - the automatic generation of feasible collision free trajectories. One of the most promising approaches to solving these problems is based on redundancy resolution control schemes, where the primary task is assigned to the trajectory tracking while the secondary task optimizes robot trajectories using various optimization goals, such as obstacle and singularity avoidance, staying within the available join limits, etc..
The basic definition of the kinematic redundancy is that the robot has more degrees of freedom than needed to accomplish the specific task. In the past, many control schemes were presented which use kinematic redundancy for the optimization of secondary tasks, such as obstacle avoidance, torque optimization, singularity avoidance, etc. All these schemes rely on a non-square Jacobian, which maps the joint velocities to the task space, which is in most cases described by the Cartesian coordinates. If the redundancy of the task can be easily described in Cartesian coordinates, i.e. the task is redundant in one of the Cartesian coordinates, then the solution is trivial an we can directly apply one of the existing control schemes. But there are tasks, such as brushing, polishing, grinding, sawing, etc. where the kinematic redundancy is hidden. It reveals when the circular shape of the tool is considered. Note that all six Cartesian coordinates are still needed to describe and to accomplish the given task.

Many authors have noticed this type of redundancy, but an efficient way how to solve it was not yet proposed (Kapoor et al., 2004; Nemec & Zlajpah, 2008; Sevier et al., 2006). As a solution to this problem we propose a virtual mechanism approach, where the tool is described as a virtual kinematic chain. In this case, the task space preserves its dimension, while the dimension of the joint space is increased by the dimension of the virtual mechanism. This approach has two major advantages. First, we can use existing robot Jacobian, which is assumed to be known. Second, the augmented part of the Jacobian, which describes the virtual mechanism, has a very simple structure in most cases. Using this formalism, we can directly apply any control and optimization algorithms developed for the kinematically redundant manipulators.
Additionally, we present some optimization procedures for secondary motion optimization. It is shown how to generate collision and wrist singularity free trajectories and how to avoid joint limits. We discuss also the case, where the given task redundancy does not allow to meet all optimization goals simultaneously. In such a case, we propose an on-line adaptation method, which reassigns a part of the primary tasks to the secondary task. The proposed approach is validated with illustrative experiments and examples - automated cell for finishing operations in the shoe production, shoe grinding cell and object tracking with the humanoid robot equipped with humanoid vision.

2. Task redundancy resolution

Robotic systems under study are n degrees of freedom (DOF) serial manipulators. We consider redundant systems, which have more DOF than needed to accomplish the task, i.e. the dimension of the joint space n exceeds the dimension of the task space m, $n > m$ and $r = n - m$ denote the degree of the redundancy. Let the configuration of the manipulator be represented by the a vector $\boldsymbol{q}_r$ of n joint positions, and the end-effector position (and orientation) by m-dimensional vector $\boldsymbol{x}_r$ of the robot tool center point positions (and orientations). The relation between the joints and the task velocities is given by the following well known expression

$$\dot{\boldsymbol{x}}_r = \mathbf{J}_r \dot{\boldsymbol{q}}_r \tag{1}$$

where $\mathbf{J}_r$ is the $m \times n$ manipulator Jacobian matrix. The solution of the above equation for $\dot{\boldsymbol{q}}_r$ can be given as a sum of particular and homogeneous solution

$$\dot{\boldsymbol{q}}_r = \bar{\mathbf{J}}_r \dot{\boldsymbol{x}}_r + \mathbf{N}_r \xi \tag{2}$$

where

$$\bar{\mathbf{J}}_r = \mathbf{W}^{-1} \mathbf{J}_r^T (\mathbf{J}_r \mathbf{W}^{-1} \mathbf{J}_r^T)^{-1}. \tag{3}$$

Here, $\bar{\mathbf{J}}_r$ is the weighted generalized-inverse of $\mathbf{J}_r$, $\mathbf{W}$ is the weighting matrix, $\mathbf{N}_r = (\mathbf{I} - \bar{\mathbf{J}}_r \mathbf{J}_r)$ is a $n \times n$ matrix representing the projection into the null space of $\mathbf{J}_r$, and ξ is an arbitrary n dimensional vector. We will denote this solution as the generalized inverse based redundancy resolution at the velocity level (Nenchev, 1989). The homogenous part of the solution belongs to the null-space of the Jacobian. Therefore, we will denote it as $\dot{\boldsymbol{q}}_n$, $\dot{\boldsymbol{q}}_n = \mathbf{N}_r \xi$.
Now consider the case where the robot Jacobian matrix $\mathbf{J}_w$ is defined in Cartesian (world) coordinate system and the dimension of the Jacobian is $6 \times n$, but the task is described in the another coordinate system, denoted with $\mathbf{p}$. The relation between the task velocities and cartesian velocities can be described as

$$\dot{\mathbf{p}} = \mathbf{J}_t \dot{\boldsymbol{x}}, \tag{4}$$

where $\mathbf{J}_t$ represents the task Jacobian. Let consider the case where the dimension of the task space m is less than the dimension of the Cartesian space, which is 6. It follows

$$\dot{\boldsymbol{x}} = \bar{\mathbf{J}}_t \dot{\mathbf{p}} + \mathbf{N}_t \mu. \tag{5}$$

Here, N_t is the 6×6 task null space matrix and μ an arbitrary 6 dimensional vector. The redundancy resolution for such case can be expressed as

$$\dot{\boldsymbol{q}}_r = \bar{\mathbf{J}}_w(\bar{\mathbf{J}}_t \dot{\mathbf{p}} + \mathbf{N}_t \mu) + \mathbf{N}_r \xi, \tag{6}$$

where vectors ξ and μ can be used for the secondary task accomplishment. The problem with the above approach is that the task Jacobian $\mathbf{J}_t$ becomes very complex even for a simple relation between then task and the cartesian coordinates. In many cases the analytical solution might even not exist.
We will demonstrate this with the shoe bottom roughing task. In the shoe bottom roughing process, we have to press the shoe bottom against a rotary grindstone and control the contact force and the contact position between the shoe bottom and the grindstone. Robot holds the shoe, while the position/orientation of the tool, grindstone in this case, is fixed. In general, the contact position on the grindstone can be freely chosen. Let define a polar task coordinates system, which describes rotary brush, as shown in the figure 1 The cartesian

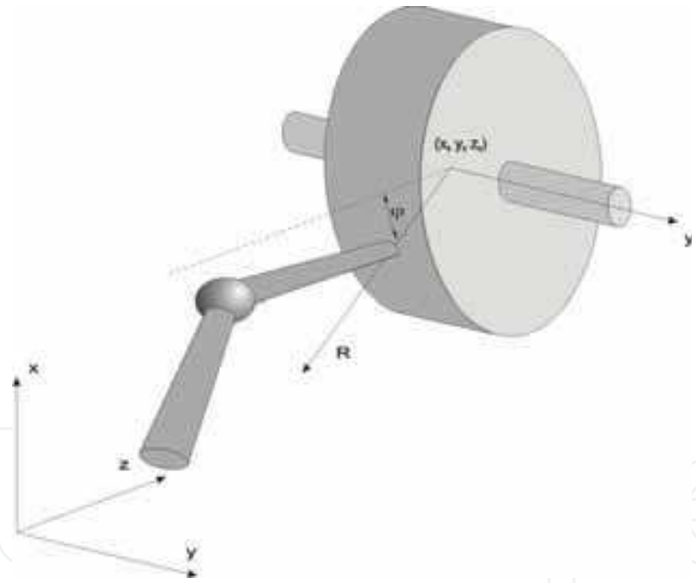

Fig. 1. Polar coordinate system of the rotary tool

coordinates are described with the $\boldsymbol{x}_r = (x, y, z, \phi, \theta, \psi)^T$ and the polar coordinates with the $\mathbf{p} = (R, y, \varphi, \phi, \nu, \psi)^T$, where ϕ, θ, ψ are the roll, pitch and yaw angles respectively, R is the radius of the polar coordinates, φ is the angle of the polar coordinate and $\nu = \theta + \varphi$. Coordinates x_0, y_0 and z_0 denote displacement of the center of the grindstone from the robot base. Obviously, coordinate φ can be freely chosen, because it does not matter which part of the rotary brush is used for the grinding. In general, also the coordinate y could be freely chosen, but since the grindstone is narrow in most cases, we will treat it as a restricted

coordinate. The resulting task Jacobian, where the third line is canceled due to the task redundancy, has the form

$$\mathbf{J}_t = \begin{bmatrix} -\dfrac{(x-x_0)}{(-z_0+z)\sqrt{\frac{\eta}{(-z_0+z)^2}}} & 0 & -\dfrac{1}{\sqrt{\frac{\eta}{(-z_0+z)^2}}} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \dfrac{-z_0+z}{\eta} & 0 & -\dfrac{x-x_0}{\eta} & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

where we used the substitute

$$\eta = (z_0 - z)^2 + (x - x_0)^2.$$

We can notice that even for the simplest case, the task Jacobian $\mathbf{J}_t$ becomes rather complex. In the case for the toroidal shaped brush we were not able to find the analytical solution of the task Jacobian using the Matlab symbolical computation toolbox.
As an alternative approach we propose to model the tool as a serial kinematic link. Let consider more general case where the robot holds the object to be machined and the work tool is fixed, as illustrated in Fig. 2. In such a case, we can define direct kinematic transformation as

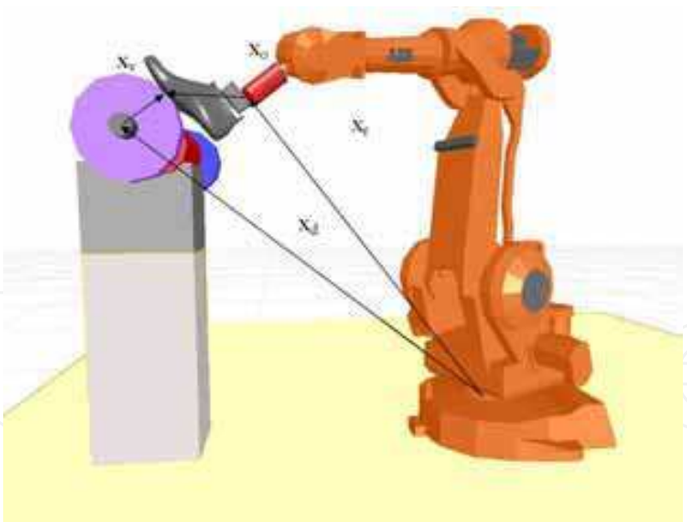

Fig. 2. The case when the robot holds an object and the work toll is fixed

$$\boldsymbol{x}_r + \left[\mathbf{R} \,\vdots\, \mathbf{I}\right] \boldsymbol{x}_o = \boldsymbol{x}_d + \boldsymbol{x}_v \tag{7}$$

where $\boldsymbol{x}_r$ is the robot Cartesian position and orientations, $\mathbf{R}$ is the robot tool rotation 3×3 dimensional matrix, $\boldsymbol{x}_o$ is the 6 dimensional vector of the object position and orientation, $\boldsymbol{x}_v$ is the 6 dimensional vector of position and orientation of the top of the virtual mechanism and

6 dimensional vector $\boldsymbol{x}_d$ describes the distance and orientation between the base coordinates system and the work tool coordinate system. Let consider robot and virtual mechanism as one mechanism with $n + n_v$ degrees of freedom, where n_v is the degree of freedom of the virtual mechanism. The configuration of the virtual mechanism can be described with the n_v dimensional vector $\boldsymbol{q}_v$. The new Cartesian position is

$$\boldsymbol{x} = \boldsymbol{x}_r - \boldsymbol{x}_v \tag{8}$$

The Jacobian of this new mechanism can be expressed as

$$\mathbf{J} = [\mathbf{J}_r \vdots -\mathbf{J}_v] \tag{9}$$

where $\mathbf{J}_r$ is the Jacobian of the robot and $\mathbf{J}_v$ Jacobian of the virtual mechanism is defined as

$$\mathbf{J}_v = \frac{\partial \boldsymbol{x}_v}{\partial \boldsymbol{q}_v}. \tag{10}$$

Note that we assume that the work tool rotation remains fixed during the execution of the task. Vector $\boldsymbol{q}_v$ of dimension n_v corresponds to the joints of the virtual mechanism.
As we can see, the task space preserves its dimension, while the joint space is increased with the dimension of the virtual mechanism. This approach has two major advantages. First, we can use existing robot Jacobian, which is assumed to be known. Second, the augmented part of the Jacobian has very simple structure in most cases. Another benefit of the proposed approach compared to the approach described with the equation 6 is that we have one larger null-space instead of two small null-spaces, which is more convenient for the trajectory optimization using the null-space motion.
As an example we present the direct kinematic transformation for the work cell shown in Fig. 5. The surface of the grinding disc is naturally described with outer surface of the torus, where R and r are the corresponding radii of the brush, as shown in the Fig. 3 and $\boldsymbol{x}$ is the task (Cartesian) coordinate of the whole system. Assuming that the robot tool position and

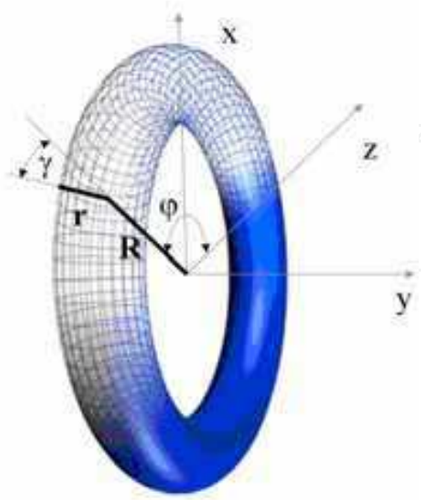

Fig. 3. Rotary brush presented as torus

robot Jacobian is known, the forward kinematics can be easily expressed as

$$\boldsymbol{x} = \boldsymbol{x}_r + \begin{bmatrix} s_\varphi\,(R + r\,c_\gamma) \\ r\,s_\gamma \\ c_\varphi\,(R + r\,c_\gamma) \\ 0 \\ -\varphi \\ \gamma \end{bmatrix}, \tag{11}$$

and the corresponding Jacobian is

$$\mathbf{J} = \begin{bmatrix} & c_\varphi\,(R + r\,c_\gamma) & -s_\varphi r\,s_\gamma \\ & 0 & r\,c_\gamma \\ \mathbf{J}_r & -s_\varphi\,(R + r\,c_\gamma) & -c_\varphi r\,s_\gamma \\ & 0 & 0 \\ & -1 & \\ & 0 & 1 \end{bmatrix}. \tag{12}$$

Here, we used the abbreviation $c_\varphi = \cos(\varphi)$, $c_\gamma = \cos(\gamma)$, $s_\varphi = \sin(\varphi)$ and $s_\gamma = \sin(\gamma)$. Note that Eq. 7 does not handle orientations correctly, since orientation vectors can not be simply added in general case. If orientations are important, we can use equation 7 for the calculation of positions only, while the orientations have to be calculated using rotation matrices as follows.

$$\mathbf{R}_o = {\mathbf{R}_v}^T \mathbf{R} \tag{13}$$

Here, $\mathbf{R}_o$ and $\mathbf{R}_v$ are 3×3 rotation matrices describing object rotation against virtual mechanism and virtual mechanism rotation expressed in the robot base coordinate system. The corresponding orientation vector can be than obtained using the transformation of the rotation matrix to the orientation vector described with euler or roll pitch yaw notation. On the other hand, orientation angles are additive for small angles. Let denote roll, pitch and yaw angles with ϕ,θ and ψ respectively. If angles are small, it holds

$$\begin{bmatrix} \frac{\Delta\phi_{rv}}{\Delta t} \\ \frac{\Delta\varphi_{rv}}{\Delta t} \\ \frac{\Delta\psi_{rv}}{\Delta t} \end{bmatrix} \simeq \begin{bmatrix} \frac{\Delta\phi_{v}}{\Delta t} \\ \frac{\Delta\varphi_{v}}{\Delta t} \\ \frac{\Delta\psi_{v}}{\Delta t} \end{bmatrix} + \begin{bmatrix} \frac{\Delta\phi_{v}}{\Delta t} \\ \frac{\Delta\varphi_{v}}{\Delta t} \\ \frac{\Delta\psi_{v}}{\Delta t} \end{bmatrix} = \mathbf{J}_r^{rot}\boldsymbol{q}_r + \mathbf{J}_v^{rot}\boldsymbol{q}_v. \tag{14}$$

Here, subscript r and v denotes the robot and the virtual mechanism respectively and *rot* denotes rotational part of the corresponding Jacobian. Equation 14 shows that even if rotation angles are not additive, velocities and thus Jacoians of the robot and virtual mechanism are additive and equation 9 hold also for the orientations. Therefore, we can directly apply any control algorithm based on Jacobian matrices and thus control and optimization algorithms developed for the kinematically redundant manipulators.

3. Control

As we mentioned in the previous section, we can directly apply any control algorithm for the kinematically redundant robot. Here we will briefly present a control law, based on the generalized inverse redundancy resolution at the velocity level in the extended operational space. Redundancy resolution at the velocity level is favorable because it enables direct implementation of the gradient optimization scheme for the secondary task. Although the control law using generalized inverse-based redundancy resolution at velocity level can not completely decouple the task and the null space (Nemec et al., 2007; Oh et al., 1998; Park et al., 2002), it enables good performance in real implementation. The joint space control law is

$$\begin{aligned}\tau_c = \quad & \mathbf{H}\mathbf{J}^T(\ddot{\boldsymbol{x}}_d + \mathbf{K}_v\dot{\boldsymbol{e}}_x + \mathbf{K}_p\boldsymbol{e}_x + \mathbf{K}_f\boldsymbol{e}_f - \dot{\mathbf{J}}\dot{\boldsymbol{q}}) + \\ & \mathbf{H}\mathbf{N}(\ddot{\boldsymbol{q}}_{nd} + \mathbf{K}_n\dot{\boldsymbol{e}}_n - \dot{\mathbf{N}}\dot{\boldsymbol{q}}) + \boldsymbol{h} + \mathbf{J}^T\boldsymbol{f}\end{aligned} \tag{15}$$

where $\bar{\mathbf{J}}$ is the inertia weighted pseudo-inverse of the Jacobian matrix $\mathbf{J}$, $\mathbf{H}$ is $n \times n$ the inertia matrix, $\boldsymbol{h}$ is n-dimensional vector of the centrifugal, coriolis and gravity forces, $\boldsymbol{F}$ is n-dimensional vector of the external forces acting on the manipulator's end effector and $\mathbf{K}_p$, $\mathbf{K}_v$, $\mathbf{K}_f$ and $\mathbf{K}_n$ are the corresponding $n \times n$ diagonal matrices with the positional, velocity, force and the null-space feedback gains. The first term of the control law corresponds to the task-space control τ_x, the second to the null-space control τ_n and the third and the fourth correspond to the compensation of the non-linear system dynamics and the external force, respectively. Here, $\boldsymbol{e}_x = \boldsymbol{x}_d - \boldsymbol{x}$ is the task-space tracking error, $\boldsymbol{e}_f = \boldsymbol{f}_d - \boldsymbol{f}$ and $\dot{\boldsymbol{e}}_n = \dot{\boldsymbol{q}}_{nd} - \dot{\boldsymbol{q}}_n$ are the force and the null-space tracking error. $\boldsymbol{x}_d$ and $\dot{\boldsymbol{q}}_{nd}$ are the desired task coordinates and the null space velocity, respectively. The details of the control law derivation can be found in (Nemec et al., 2007).

An attention should be paid on the selection of the inertia of the virtual link. The inertia matrix $\mathbf{H}$ has the form

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_m & 0 \\ 0 & \mathbf{H}_v \end{bmatrix} \tag{16}$$

where $\mathbf{H}_m$ is the manipulator inertia matrix and $\mathbf{H}_v$ is the diagonal matrix describing the virtual mechanism inertia. Clearly, $\mathbf{H}_v$ can not be zero, but arbitrary small values can be chosen describing the lightweight virtual mechanism. Selection of the inertia matrix of the virtual mechanism affects only the null space behavior of the whole system. Heavy virtual links with high inertia will slow down the movements of the virtual links. Therefore, low inertia of the virtual links makes suitable choice. On contrary, we can assume that the virtual links have no gravity and no coriolis and centrifugal forces and the corresponding terms in the vector $\boldsymbol{h}$ can be set to zero. Control law 15 assumes the feedback from all joints, including non-existing virtual joints. There are multiple choices how to provide the joint coordinates and the joint velocities of the virtual link. A suitable method is to build a simple model composed of a double integrator

$$\begin{aligned}\dot{\boldsymbol{q}}_v &= \int \mathbf{H}_v^{-1}\tau_{cv} \\ \boldsymbol{q}_v &= \int \dot{\boldsymbol{q}}_v\end{aligned} \tag{17}$$

where τ_{cv} is the part of the control signals corresponding to the virtual link.

4. Null space motion determination trough optimization

As we mentioned previously, one of the main problems in automatic trajectory generation is the inability to assure that the generated trajectory is feasible using a particular robot, either because of possible collisions with the environment or because of the limited workspace of the particular robot. Limitations in the workspace are usually not subjected to the tool position, but rather to the tool orientation. Another sever problem are wrist singularities, which can not be predicted in the trajectory design phase on a CAD system. A widely used solution in such cases is off-line programming with graphical simulation, where such situation can be detected in the design phase of the trajectory. Unfortunately this is a tedious and time consuming process and therefore not applicable in customized production, where almost each work piece can vary from the previous one (Dulio & Boer, 2004; Nemec & Zlajpah, 2008). The problem can be efficiently solved using the kinematic redundancy and null space motion, which changes the robot configuration, but does not affect the task space motion. The force and the position tracking are of the highest priority for a force controlled robot and are therefore considered as the primary task. The secondary task can be defined as a result of the local optimization of a given cost function. Here we will use the gradient projection technique, which has been widely implemented for the calculation of the null space velocity that optimizes the given criteria. The reason for this is that a variety of performance criteria can be easily expressed as gradient function of joint coordinates.
Let be the desired cost function, which has to be maximized or minimized. Then the velocities

$$\dot{\boldsymbol{q}}_n = \mathbf{N}\mathbf{H}^{-1}\frac{\partial p}{\partial \boldsymbol{q}}k, \tag{18}$$

maximize cost function for any $k > 0$ and minimize cost function for any $k < 0$ (Asada & Slotine, 1986), where k is an arbitrary scalar which defines the optimization step. In our case we have selected a compound p which maximizes the distances between obstacle and the robot links or robot work object, maximizes the distance to the singular configuration of the robot and maximizes the distance in joint coordinates between current joint angle and joint angle limit.
For the obstacle avoidance we use approach based on the potential field pointing away from the obstacle as illustrated in the figure 4 (Khatib, 1986; 1987).

$$p_a = \frac{1}{2}\boldsymbol{d}_i^T\boldsymbol{d}_i \tag{19}$$

where $\boldsymbol{d}_i$ is the shortest distance between the obstacle and the robot body. In our case the desired objective is fulfilled if the imaginary force is applied only on the robot joints and we can obtain the cost function gradient in a simple form as

$$\frac{\partial p_a}{\partial \boldsymbol{q}} = \boldsymbol{d}_1^T\mathbf{J}_{0,1}^{pos} + \boldsymbol{d}_2^T\mathbf{J}_{0,2}^{pos} + ... + \boldsymbol{d}_{n-1}^T\mathbf{J}_{0,n-1}^{pos}, \tag{20}$$

where $\mathbf{J}_{0,i}^{pos}$ denotes Jacobian matrices between base (the first index in the subscript) and i-th joint (the second index in the subscript) regarding the robot positions only.

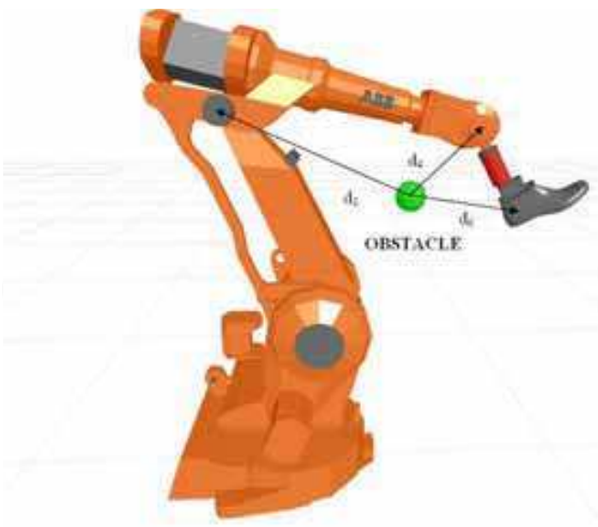

Fig. 4. Obstacle avoidance using potential field approach

The cost function for the joint limits avoidance is defined as (Chaumette & Marchand, 2001; Nemec & Zlajpah, 2008)

$$p_l = \frac{1}{2} \begin{Bmatrix} (q_{max} - q)^2, |q_{max} - q| < \epsilon \\ 0 \\ (q_{min} - q)^2, |q_{min} - q| < \epsilon \end{Bmatrix} \tag{21}$$

where ϵ is a positive constant defining the neighborhood of joint limits. Cost function gradient for obstacle avoidance is thus

$$\frac{\partial p_l}{\partial \boldsymbol{q}} = \boldsymbol{q}_{lim} - \boldsymbol{q}, \tag{22}$$

where $\boldsymbol{q}_{lim}$ denotes closed joint limits, that can be either $\boldsymbol{q}_{max}$ or $\boldsymbol{q}_{min}$.
For the singularity avoidance we use the manipulability index defined as (Asada & Slotine, 1986)

$$p_s = \sqrt{|\mathbf{J}\mathbf{J}^T|}, \tag{23}$$

where the gradient can be expressed as (Park et al., 1999)

$$\frac{\partial p_s}{\partial \boldsymbol{q}} = p_s[trace(\frac{\partial p_s}{\partial q_1}\bar{\mathbf{J}}) \quad trace(\frac{\partial p_s}{\partial q_2}\bar{\mathbf{J}}) \quad ..... \quad trace(\frac{\partial p_s}{\partial q_n}\bar{\mathbf{J}})]. \tag{24}$$

Note that in most cases the singularity is due to the spherical wrist of the robot and the equation 24 can be reduced by taking into the consideration only the last three joints which correspond to the wrist movement. Unfortunately, the partial derivative $\frac{\partial p_s}{\partial \boldsymbol{q}}$ is not easy to calculate. However, we can use the numerical derivative of the manipulability measure instead.
In practice we deal only with one optimization goal at the time, depending on which secondary task is the most critical one. Note that simple addition of different optimization gradients my not be appropriate, although it is often used in practice. Namely, it could

happen that two optimizations gradients act in opposite, e.g. obstacle avoidance might push joints toward the physical limits.
It might happen that there is no enough redundancy to fulfil the required secondary task. It the secondary task is critical, like obstacle avoidance or joint limit avoidance, the task execution has to be interrupted. The only possibility in such a case is to modify the primary task. For some tasks like polishing it is not necessary to assure the strict orientations of the tool. Therefore we can define orientation tracking as a secondary task. Using this approach we obtain additional degrees of redundancy, which can be successful used for the accomplishment of the critical secondary task. In such a case we have to assure that the tool orientation follows the desired orientation whenever it is possible. To do this, we define orientation tracking as an optimization procedure, where we minimize the difference between the desired orientation vector $\boldsymbol{x}_{od}$ and actual orientation vector $\boldsymbol{x}_o$.

$$p_o = \frac{1}{2}(\boldsymbol{x}_{od} - \boldsymbol{x}_o)^T(\boldsymbol{x}_{od} - \boldsymbol{x}_o) \tag{25}$$

$$\frac{\partial p_o}{\partial \boldsymbol{q}} = -(\boldsymbol{x}_{od} - \boldsymbol{x}_o)^T \mathbf{J}^{rot} \tag{26}$$

The dimension of the vector $\boldsymbol{x}_o$ depends from the specific application depending on how much redundancy we need in order to accomplish the most critical secondary task. $\mathbf{J}^{rot}$ denotes the corresponding rotational part of the Jacobian which relates to the selected components of $\boldsymbol{x}_o$.

5. Shoe grinding example

In the shoe assembly process, in order to attach the upper with the corresponding sole, it is necessary to remove a thin layer of the material off the upper surface so that the glue can penetrate the leather. To do this, the robot has to press the shoe against the grinding disc with the desired force while executing the desired trajectory. In the past, there were several approaches how to automate this operation. For mass production, there are special NC machines available. Their main drawback is relatively complicated setup and are therefore not suitable for the custom made shoes. Required flexibility is offered by the robot based grinding cell. In the EUROShoeE project (Dulio & Boer, 2004), a special force controlled grinding head has been designed. The robot manipulated with the grinding head while the shoe remained fixed on the conveyor belt (Jatta et al., 2004) The main drawback of this approach is relatively heavy and expensive grinding head. Additionally, force control can be applied only in one direction. In our approach, the robot holds the shoe and presses it against the grinding disc of a standard grinding machine as used in the shoe production industry. The impedance force control was accomplished by the robot using universal force-torque sensor mounted between the robot wrist and the gripper which holds the shoe last. It is well known that the kinematic redundancy enables greater flexibility in execution of complex trajectories. For example, also humanoid hand dexterity is subjected by its kinematical redundancy. We used Mitsubishi Pa10 robot with 7 D.O.F in our roughing cell, which has one degree of redundancy. Additional two degrees of redundancy were obtained by treating the grinding disc as a virtual mechanism. The surface of the grinding disc can be naturally described with the outer surface of the torus, where R and r are the corresponding radius of the grinding disc, as shown in the Fig. 3. Thus we have 9 degrees of freedom, 6 of them are required to describe the grinding task, while the remaining three degrees of freedom are used for the obstacle avoidance, joint limits avoidance and singularity avoidance. The prototype of the

Fig. 5. Experimental cell for shoe bottom roughing

cell is shown in figure 5. It consists of the Mitsubishi Pa10 robot with a force/torque sensor Jr3 mounted in the robot wrist, a grinding machine, a Pa10 robot controller and a cell control computer, which coordinates the task and calculates the required robot torques. The control computer is connected to the robot controller using ArcNet. The frequency rate of the control algorithm (Eq. 15) and the motion optimization algorithm (Eq. 18) is 700 Hz. The grinding path is obtained from CAD model of the shoe. For this purpose, the control computer is connected to the shoe database computer using Ethernet. Unfortunately, CAD model itself can not supply all necessary data for the grinding process. CAD models are usually available for the reference shoe size, therefore, non-linear grading of the shoe shape is necessary for the given size. Additionally, some technological parameters such as material characteristics and shoe sole gluing technology have to be taken into account during the grinding trajectory preparation. For this purpose, we have developed a special CAD expert program, which enables the operator to define additional technological and material parameters. The program then automatically generates the grinding trajectory. In order to show the efficiency of the proposed algorithm, we defined the shoe grinding trajectory as seen in the Fig 4. Note that without using trajectory optimization is is very hard to execute the given task without splitting the desired trajectory in two or more fragments. Fig 5 shows how the system rotated joints of virtual mechanism in order to avoid the joint limits and to minimize joint velocities of the robot and virtual mechanism.

6. Automation of finishing operations in shoe assembly

Finishing operations in shoe manufacturing process comprises operations such as application of polishing wax, polishing cream and spray solvents, and brushing in order to achieve high gloss. These operations require skilled worker and are generally difficult to automate due to the complex motion trajectories. The finishing cell consists of the shoe polishing machine, machine for application of polishing creme, spray cabin for application of the polishing solvents and an industrial robot, as seen in Fig 4. The 6. d.o.f robot is a commercially available product from ABB, rest of the cell components were not available and had to be developed

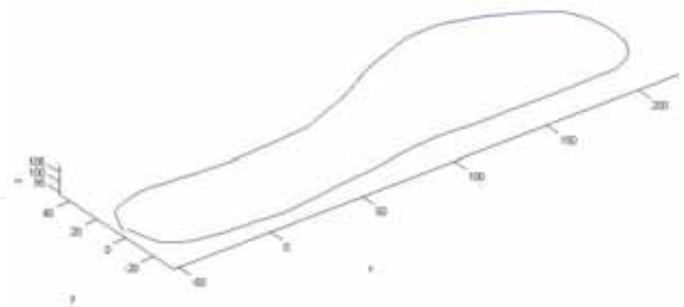

Fig. 6. Shoe grinding trajectory

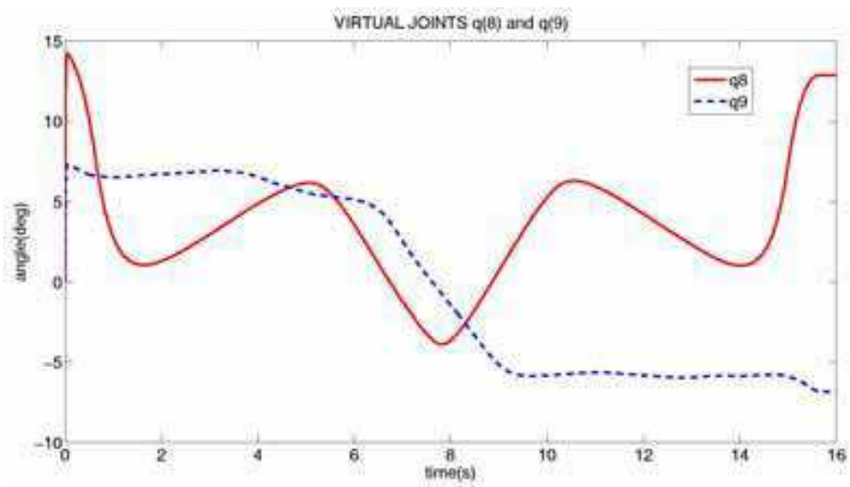

Fig. 7. Virtual mechanism angles q8 and q9

especially for this purpose. Customized mass production differs from the mass production because virtually any product item can differ from the previous one. Therefore, manual teaching and manual preparation of the manufacturing programs is not acceptable. The customized mass production requires that all production phases are prepared in advance during the design phase of the specific shoe model. Modification of the part programs for the specific shoe model, required for the customization, has to be done automatically without any human intervention. Therefore, new CAD tools for finishing operations had to be developed. One of the main problems in automatic trajectory generation is the inability to assure that the generated trajectory is feasible using a particular robot, either because of the possible collisions with the environment or because of the limited workspace of the particular robot. Limitations in the workspace are usually not subjected to the tool position, but rather to the tool orientation. Another sever problem are wrist singularities, which can not be predicted in the trajectory design phase on a CAD system. A widely used solution is such case is off-line programming with graphical simulation, where such situation can be detected in the design phase of the trajectory. Unfortunately this is a tedious and time consuming process and therefore not applicable in customized production, where almost each work piece can vary from the previous one (Dulio & Boer, 2004). The problem was efficiently solved using the trajectory optimization based on kinematic redundancy of the manipulator (Nemec &

Fig. 8. Finishing cell

Zlajpah, 2008). For a given task, the obstacle avoidance can be accomplished only if the robot is redundant. Note that the degree of redundancy depends on the task the robot is performing. For example, a 6 D.O.F robot is kinematicaly redundant for spraying and creaming operations in shoe production. Due to the circular shape of the cream application brush and spray beam, roll angle or the robot is free to choice. For brushing operations, there is another type of redundancy due to the circular shape of felt rollers, as illustrated in Fig. 1. Namely, the tool centre point is not restricted to be a fixed point, rather it can be freely chosen at the circumference of the tool. Unfortunately, in general one degree of redundancy is not enough to satisfy simultaneously all secondary tasks - obstacle avoidance, singularity avoidance and preserving the joint angles within their physical limits. More flexibility is offered by the fact that for some tasks it is not necessary to assure strict orientations of the tool. This can be interpreted as two additional degrees of redundancy. In robot trajectory generation, we define primary and secondary task. Primary task is the position of the TCP of the robot. We have multiple secondary tasks, such as a) Maximizing the distance between the robot joins and

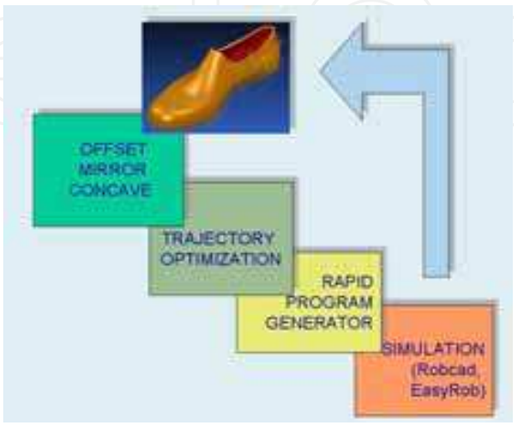

Fig. 9. Batch trajectory generation

the environment objects-obstacles. This task prevents the robot to collide with the obstacles b) Maximizing the distance between the joint position and join limits. This task prevents the robot to come to the join limits. c) Maximizing the distance between the actual and singular pose, which avoids wrist singularity d) Minimizing the difference between the desired and actual tool orientation. Secondary tasks generate robot tool orientation based on the gradient optimization in Jacobian null-space. Since we did not have access to the low level robot control, we implemented virtual mechanism approach as a batch procedure in the trajectory optimization module, as illustrated in the figure 9. Due to the physical limitations of the robot it is possible that the procedure does not converges. In such a case the optimization stops and off-line programming system is used to check and verify the robot configuration and the desired task trajectory. In most cases after the successful accomplishment of the trajectory optimization the verification with off-line programming system is not necessary and the trajectory can be downloaded directly to the robot controller.

7. Humanoid head control

The task of the robot head (Figure 10) is to keep the object in the center of both narrow-angle camera images. The head has to assure proper gaze direction of both eyes (cameras). Therefore, the task has four DOFs, since the gaze direction of each eye is defined by two parameters. A humanoid head typically has more DOFs, e. g. seven on the head of Fig. 10 and the degree of redundancy is three. Gaze direction is a function of a 3-D point in space as well as the position of the eye (see Figure 10). When the head is moved, the position of the eye change and that affects the gaze direction. Thus the task is a function of a point in space as well as a function of the head configuration. The above statement of the problem is not the most common way to describe a task - in general, a task is not a function of a robot configuration. To solve such problems in a systematic way, we have used the virtual mechanism approach (Omrcen & Ude, 2009). Let us explain the virtual mechanism approach on a simple pointing example, where finger points to an object. The task has one DOF and can be defined as an angle of the line from the finger to the object. If the object moves, the angle has to change in order to maintain the correct pointing direction. Similar, when the hand moves, the angle also has to change. The task is therefore a function of the hand and the object position. By adding a virtual prismatic link to the finger, we require that the extension of the finger touches the object. The system now has two DOFs (the angle of the finger and the length of the virtual mechanism). Using that notation, the task can be described as a positioning task; the end of the virtual link has to touch the object. The task now has two DOFs and the description of the task is now only the position of the object and not a function of the hand position. The introduction of the virtual mechanism increases the DOFs of the mechanism by one; however, it also increases the degree of the task. The degree of redundancy remains the same, while the description of the task has now been simplified and systematized: instead of specifying the desired pointing direction, we can consider the problem as a classic inverse kinematics task. In the case of a robot head we define a new virtual mechanism in each eye that points from the robot's eye to a point in space (see Figure 11). Virtual mechanism is a prismatic link, which adds a new degree of freedom to the system. The length of the new link is the distance from the eye to the 3-D point in space. Due to the two new DOFs added to the system, the system now has nine DOFs (one additional per each eye). However, the degree of the task has also increased. The task is not defined as a gaze direction of each eye but as the positioning of the end of the virtual link. Instead of controlling the gaze direction, the task is simplified to a simple

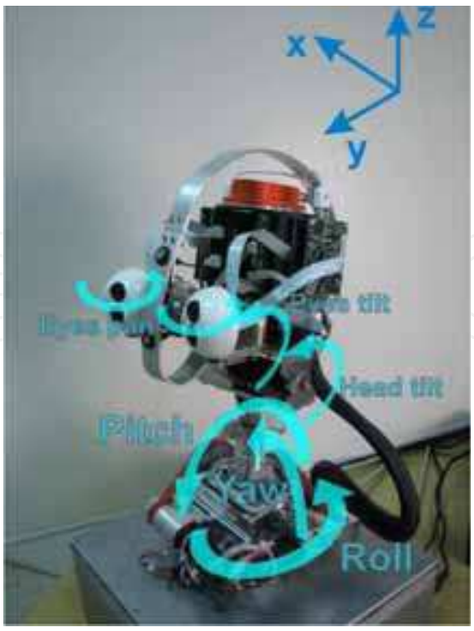

Fig. 10. Humanoid head

position control in space. Degree of redundancy should and does remain the same. Figure 11 schematically shows the kinematics of the head with additional virtual mechanisms.

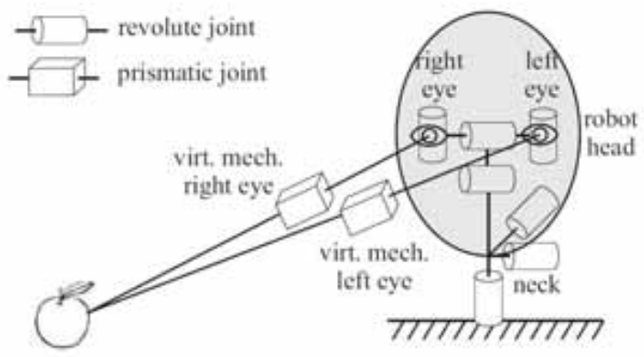

Fig. 11. Schematics of humanoid head with virtual mechanisms

8. Conclusion

In the chapter we deal with the automatic trajectory generation for industrial robots. Automatically generated programs have to consider various limitations of the robot mechanism, such as joint limits, wrist singularity and possible collisions of the robot with the environment. The required flexibility required to solve the above problems is offered by the kinematic redundancy. We proposed a new method of solving the kinematic redundancy which

arises from the shape of the work tool. The main benefit of the proposed method is the simplicity and efficiency. It can be used on the existing robot controllers with very moderate changes of the control algorithm. The proposed approach is particularly efficient for the tasks which require automatic trajectory generation, since it helps to generate fault-tolerant trajectories The proposed approach was implemented in various industrial and non-industrial applications. We have outlined three illustrative examples : shoe bottom roughing task, automation of finishing operations in shoe assembly and control of a humanoid head. Two possible modes of implementation were proposed. Implementation of the proposed approach in the control loop allows real time optimization procedure, e.q. obstacle avoidance and to accomplish the given task at the same time. Another, perhaps for the practical implementation even more attractive possibility is to use the proposed approach in the trajectory generation module rather than in the control algorithm. Doing so, we get benefits of the kinematic redundancy due to circular shape of the tool without any modification of the robot controller. This latter approach was successfully implemented in the cell for custom finishing operations in shoe assembly. Unfortunately, in this case we have to deny to the real-time trajectory modifications, which can be only implemented in the control loop.

9. References

Asada, H. & Slotine, J.-J. (1986). *Robot Analysis and Control*, John Wiley & Sons.

Chaumette, F. & Marchand, . (2001). A redundancy-based iterative approach for avoiding joint limits: Application to visual servoing, *IEEE Transactions on Robotics and Automation, 17(5)*.

Dulio, S. & Boer, S. (2004). Integrated production plant (ipp): an innovative laboratory for research projects in the footwear field, *Int. Journal of Computer Integrated Manufacturing, 17(7) : 601-611*.

Jatta, F., Zanoni, L., Fassi, I. & Negri, S. (2004). A roughing/cementing robotic cell for custom made shoe manufacture, *Int. J. Computer Intergrated Manufacturing, 17(7) : 645–652*.

Kapoor, C., Pholsiri, C. & Tesar, D. (2004). Manipulator task-based performance optimization., *Proc of DECT'04 ASME Conference, Salt Lake City*.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots, *Int. J. of Robotic Research, 5 : 90 – 98*.

Khatib, O. (1987). A unified approach for motion and force control of robot manipulators:the operational space formulation, *IEEE Trans. on Robotics and Automation, 3(1) : 43 – 53*.

Nemec, B. & Zlajpah, L. (2008). Robotic cell for custom finishing operations, *Int. J. Computer Intergrated Manufacturing, 21(1) : 33–42*.

Nemec, B., Zlajpah, L. & Omrcen, D. (2007). Comparison of null-space and minimal null-space control algorithms, *Robotica, 2007, 25(5):511–520*.

Nenchev, D. N. (1989). Redundancy resolution through local optimization: A review, *J. of Robotic Systems, 6(6) : 769 – 798*.

Oh, Y., Chung, W., Youm, Y. & Suh, I. (1998). Experiments on extended impedance control of redundant manipulator, *Proc. IEEE/RJS Int. Conf. on Intelligent Robots and Systems, : 1320 – 1325, Victoria*.

Omrcen, D. & Ude, A. (2009). Redundancy control of a humanoid head for foveation and 3-d object tracking: A virtual mechanism approach., *submitted to Journal of Advanced Robotics*.

Park, J., Chung, W. & Youm, Y. (1999). Computation of gradient of manipulability for kinematically redundant manipulators including dual manipulators system, *Transactions on Control, Automation and Systems Engineering, 1(1).*

Park, J., Chung, W. & Youm, Y. (2002). Characterization of instability of dynamic control for kinematically redundant manipulators, *Proc. IEEE Conf. Robotics and Automation, : 2400 – 2405, Washington DC*.

Sevier, J., Kapoor, C. & Tesar, D. (2006). Benefitting from underutilized task specific resources for industrial robotic systems, *International Symposium on Robotics and Applications (ISORA) Budapest*.

IntechOpen

Contemporary Robotics - Challenges and Solutions

Edited by A D Rodi

ISBN 978-953-307-038-4
Hard cover, 392 pages
Publisher InTech
Published online 01, December, 2009
Published in print edition December, 2009

This book book is a collection of 18 chapters written by internationally recognized experts and well-known professionals of the field. Chapters contribute to diverse facets of contemporary robotics and autonomous systems. The volume is organized in four thematic parts according to the main subjects, regarding the recent advances in the contemporary robotics. The first thematic topics of the book are devoted to the theoretical issues. This includes development of algorithms for automatic trajectory generation using redudancy resolution scheme, intelligent algorithms for robotic grasping, modelling approach for reactive mode handling of flexible manufacturing and design of an advanced controller for robot manipulators. The second part of the book deals with different aspects of robot calibration and sensing. This includes a geometric and treshold calibration of a multiple robotic line-vision system, robot-based inline 2D/3D quality monitoring using picture-giving and laser triangulation, and a study on prospective polymer composite materials for flexible tactile sensors. The third part addresses issues of mobile robots and multi-agent systems, including SLAM of mobile robots based on fusion of odometry and visual data, configuration of a localization system by a team of mobile robots, development of generic real-time motion controller for differential mobile robots, control of fuel cells of mobile robots, modelling of omni-directional wheeled-based robots, building of hunter- hybrid tracking environment, as well as design of a cooperative control in distributed population-based multi-agent approach. The fourth part presents recent approaches and results in humanoid and bioinspirative robotics. It deals with design of adaptive control of anthropomorphic biped gait, building of dynamic-based simulation for humanoid robot walking, building controller for perceptual motor control dynamics of humans and biomimetic approach to control mechatronic structure using smart materials.

Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen

IntechOpen