Chapter 52
Path Modelling and 3D Robot Visualization for Model-Based Control of Articulated Robots

Květoslav Belda and Karel Dvořák

Abstract The paper deals with the three-dimensional (3D) path modelling and visualization used in the simulation of model predictive control design of industrial robot manipulators. A possible comprehensive procedure is introduced as a specific sequence of the following points: (i) path modelling using set of control points with G-code generation and time parametrization; (ii) construction of a photo-realistic 3D virtual reality robot model including surrounding workplace; (iii) 3D robot visualization using Virtual Reality Modeling Language (VRML V2.0); (iv) brief overview of model predictive control design; (v) simulation of the control with visualization using prepared 3D virtual reality robot model. This sequence is applied to the one representative of articulated robots—ABB robot IRB 140. Siemens NX Software (points i–iii) and MATLAB/Simulink environment (points iii–v) are proposed as suitable work tools.

Keywords Siemens NX software · MATLAB/Simulink 3D animation toolbox · Model predictive control · Articulated robots · Virtual reality

K. Belda (✉)
Department of Adaptive Systems,
The Czech Academy of Sciences, Institute of Information Theory and Automation, Pod Vodárenskou věží 4, 180 00 Prague 8, Czech Republic
e-mail: belda@utia.cas.cz

K. Dvořák
Department of Technical Studies, The College of Polytechnics Jihlava,
Tolstého 16, 586 01 Jihlava, Czech Republic
e-mail: karel.dvorak@vspj.cz

E. Zattoni et al. (eds.), *15th European Workshop on Advanced Control and Diagnosis (ACD 2019)*, Lecture Notes in Control and Information Sciences - Proceedings,
https://doi.org/10.1007/978-3-030-85318-1_52

52.1 Introduction

Path modelling is a necessary preparatory task of every application including motion control in general. To ensure effecient path modeling and verification before its run in specific motion control system, appropriate modeling and simulation tools should be used. In this paper, we will consider two suitable tools: Siemens NX for construction and modelling and MATLAB/Simulink for motion control simulation with 3D realistic visualization.

Let us firstly introduce mentioned tools in view of proposed path modelling and simulation and visualization of robot model-based motion control. Siemens NX is a Computer Aided (CAx) application that includes a wide portfolio of tools for modelling, simulations, and production technologies such as Computer Aided Design (CAD) or Computer Aided Manufacturing (CAM).

The basic, default output of NX is a virtual prototype that is in the form of an individual model or assembly. A virtual prototype represents a digital representation (digital model) of a real product (final component, product).

A starting point of the modelling is parametric or non-parametric geometry that forms the appropriate digital model. This model as default, usually exact CAD model or assembly, is used for subsequent procedures as associative or non-associative copies. For associative copies, the basic geometry is controlled by just the default model. The geometry can be simplified for simulation purposes, e.g. by removing insignificant details that do not affect the resulting, simulated tracking parameters. However, such simplification can often reduce the computational demands of simulation. This approach, usually referred to as a common model method, digital twin, or Master Model Concept, e.g. in [2, 7], is illustrated in Fig. 52.1.

To support the simulation of the robot control, 3D models of the main robot components with corresponding shape and dimensional accuracy are used. Parametric setting of the assembly allows kinematics to be controlled through a set of variables or by external tools. This feature is useful for the visualization of simulation of robot behaviour. The basic kinematic manipulation during construction in NX is realized through a manipulation tool, where the vector and the subsequent translation or rotation of the specified parameter can be changed.

For manipulation, it is also possible to use individual geometric objects of the model or assembly or a dynamic manipulation tool in general. Furthermore, the selected parameters can be directly determined as well as movements of the specified parts or points on the parts of the mechanism can be analysed by means of the appropriate pre-configuration of Assembly Constraints such as distance, angle, collinearity, and perpendicularity [12].

The aforementioned approach is helpful for the construction of 3D CAD models. Note that for construction of a dynamic model of the robot (i.e. equations of motions), necessary for model-based control, NX is also a helpful tool since it can offer informative parameters such as volumes or moments of inertia relating to specific densities of individual robot components. However, for the design and development of advanced model-based control, NX cannot offer adequate flexibility due to its strict purpose-oriented industrial functionality.

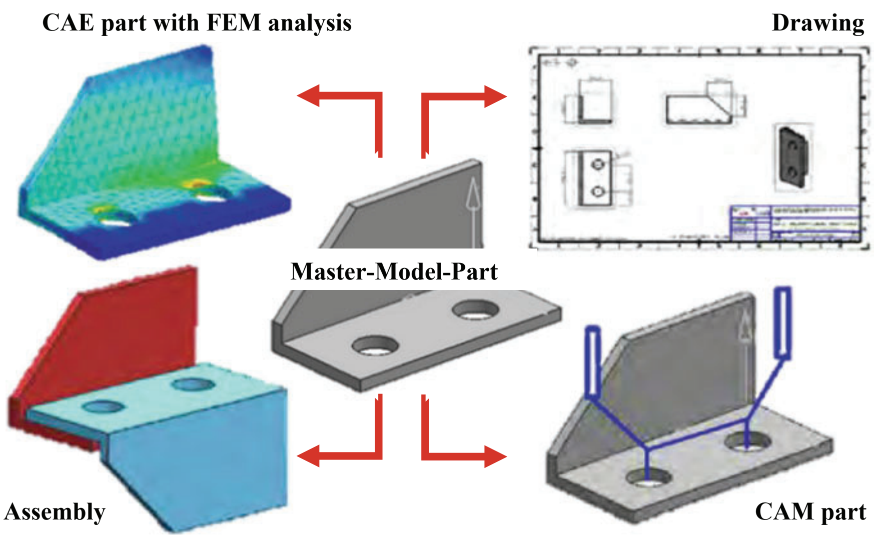


Fig. 52.1 Master model concept

For control design, the MATLAB/Simulink environment represents an adequate tool, where control algorithms of developed control approach can be straightforwardly formulated [5]. Furthermore, obtained 3D model from NX can be involved and used for 3D visualization together with simulation of the control. Thus, the usual development of control algorithms can be connected to 3D visualization through Simulink 3D Animation Toolbox. Then, user can obtain a realistic demonstration of a run of a real robot including dynamic behaviour corresponding to control actions generated from control algorithms. One example of such complex model-based control approaches is Model Predictive Control (MPC) that represents a promising way for future industrial control systems. For 3D visualization, Virtual Reality Modeling Language (VRML V2.0) is used.

The paper is organized as follows. Section 52.2 deals with path modelling. Section 52.3 will explain construction of a 3D virtual robot model. Section 52.4 will describe 3D robot visualization in MATLAB/Simulink. Finally, Sect. 52.5 will demonstrate simulation of robot MPC with simultaneous 3D robot visualization. The corresponding conceptual designing diagram is shown in Fig. 52.2.

52.2 Path Modelling in 3D

In our explanation, let us consider a geometric path, described by G-code. Such a path will represent an ideal, geometrically accurate, curve. The curve should smoothly connect geometric centres of the tool path, where the tool is fixed to the robot flange.

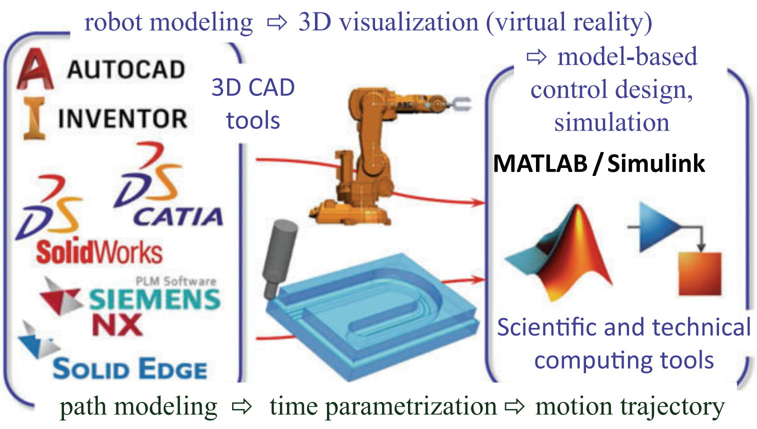

robot modeling ⇨ 3D visualization (virtual reality)
⇨ model-based control design, simulation
AUTOCAD
INVENTOR
3D CAD tools
CATIA
SolidWorks
PLM Software SIEMENS NX
SOLID EDGE
MATLAB / Simulink
Scientific and technical computing tools
path modeling ⇨ time parametrization ⇨ motion trajectory

Fig. 52.2 Conceptual designing diagram

The G-code is standard for the description of the paths of machines in industrial production. Here, let us consider for simplicity only several basic instructions such as `G00`, `G01`, `G02`, and `G03` representing rapid positioning, linear interpolation, circular clockwise/counterclockwise interpolations, respectively.

Time parametrization is a necessary operation that is involved in the interpretation of the G-code. The G-code uniformly represents (describes) motion path and can contain also simple pieces of information about kinematic parameters such as motion velocities (feed rates) or time dwells.

To execute the code in a particular production machine, it is necessary to have a suitable algorithm that is able to interpret the individual lines of the G-code program for the control system. In the majority of cases, it means execution of specific time parametrization, the result of which is ordered couples of time and appropriate coordinate represented data for the control system. These data are fed within the control system in specific synchronous time sampling (time period).

A simple example of a CNC program, considered in this paper taking NX into account, is listed in Table 52.1.

52.3 3D Virtual Robot Model

To obtain a photo-realistic 3D virtual robot model, let us consider NX for initial individual positioning and orientation of robot components in the 3D space. The adequately, specifically positioned and oriented components will be used in the creation of 3D visualization.

The main structural objects of individual 3D models are points, curves, areas, and surface areas. Furthermore, the position and orientation of the coordinate sys-

Table 52.1 G-code of testing trajectory (*mm*)

N010	G00	X500	Y0	Z400	G17		
N020	G03	X600	Y100	Z400	I0	J100	K0
N030	G02	X700	Y200	Z400	I100	J0	K0
N040	G01	X800	Y200	Z400			
N050	G00	X700	Y200	Z400			
N060	G00	X700	Y200	Z500			
N070	G00	X800	Y200	Z500			
N080	G01	X700	Y200	Z500			
N090	G03	X700	Y-200	Z500	I0	J-200	K0
N100	G01	X800	Y-200	Z500			
N110	G00	X700	Y-200	Z500			
N120	G00	X700	Y-200	Z400			
N130	G01	X800	Y-200	Z100			
N140	G00	X700	Y-200	Z400			
N150	G02	X600	Y-100	Z400	I0	J100	K0
N160	G03	X500	Y0	Z400	I-100	J0	K0
N170	G00	X400	Y0	Z400	M30		

(considered starting point: X400 Y0 Z400)

tem are important for construction as well. Both parameters determine the basic reference of the model, e.g. with respect to the higher assembly, or for determining the simulation characteristics.

The models are located in an assembly where their position can be defined by means of the location of their local coordinate systems, or by the assembly constraints. The constraints mean the conditions of the mutual positioning of objects. Figure 52.3 illustrates such local coordinate systems. They are associated with individual robot objects (bodies) or their characteristic points as an intersection of symmetry axes or rotational axes.

Conditions can be defined for local coordinate systems or for geometric objects of components—models. For simulation management purposes, it is optimal if assembly constraints correspond to the functional characteristics of real connections, which supports simulation and analysis of real behaviour. The position and orientation are determined primarily by the absolute Coordinate SYStem (absolute CSYS). Its position and orientation are invariable.

The position and orientation of the geometric objects of the model are defined for the absolute CSYS, and the visual manipulation by the model represents the movement around the coordinate system. The absolute CSYS is also usually a portable reference when exporting curve or bulk model objects to “third-party” formats.

For the models of virtual prototypes of kinematic mechanisms that represent the robot model, it is appropriate to configure the object of the model with respect to the absolute CSYS whose position is either in the centre of the component body

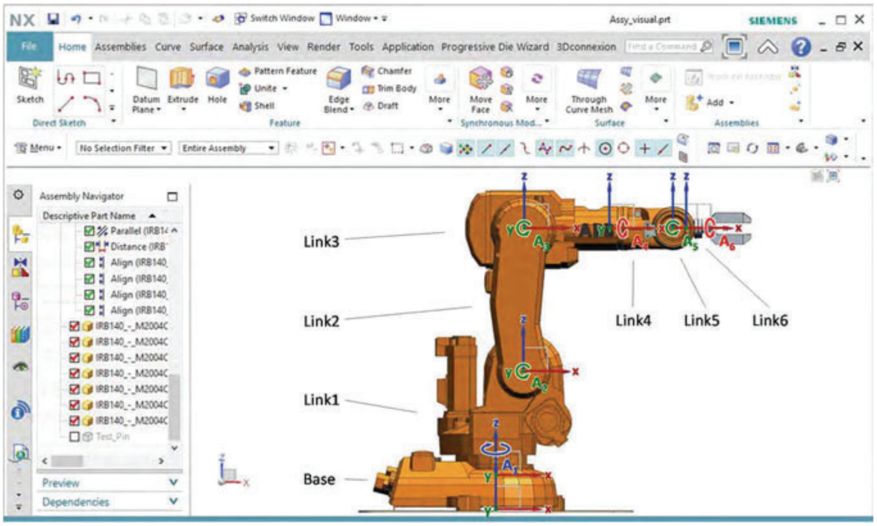


Fig. 52.3 3D VR model of ABB robot IRB 140 in NX environment

(assembly) or in the kinematic node. The kinematic node can be, as was already indicated, a point on the rotational axis at the intersection with the moving plane of the adjacent component. It may be a point on the edge, area, or body volume, or a virtual point outside the body.

Another local coordinate system is the Work Coordinate System (WCS). WCS is only relevant for editing a custom component model. It is not an object; it is a virtual, non-associative temporary coordinate system that allows localization of the position of the object at the beginning and the directions of the vectors to the main axes or to the XC axis in the XC-YC plane.

An important reference object is the user-defined coordinate system Date CSYS [10]. It is an element composed of the basic reference objects—the three reference axes, the planes between the axes, and the reference point. There may be an arbitrary number of user-defined coordinate systems in the model.

In addition to the local significance for component modelling, it is advantageous to have one Datum CSYS based and aligned with the absolute coordinate system of the model. The Datum CSYS, including reference geometry, is usable in the context of the assembly to define the mutual positioning conditions of individual components. A typical example is the collinearity of some of the axes of the two components to provide rotation around the axis and the subsequent definition of the angle of rotation by the parameter—user-defined variable, or the distance in the direction of the defined vector between the selected objects.

The parametric setting of the assembly allows kinematics to be controlled through a set of variables by means of external tools, where control parameters can be generated according to a specific point tracking requirement or based on a kinematic simulation. A set of points for subsequent kinematic analysis and optimization can be obtained using other diagnostic tools [13].

Individual CSYS are shown in Fig. 52.3. A photo-realistic model for 3D visualization is obtained by export of the 3D model in NX to VR World `*.wrl` format of Virtual Reality Modeling Language (VRML V2.0).

52.4 Setup of 3D Visualization

3D virtual robot model, serving for online 3D visualization during robot motion control, is constructed in 3D World Editor of MATLAB/Simulink 3D Animation Toolbox. This section introduces the structure setup of the VR model and direct and inverse kinematic transformations. The transformations are needful for the determination of the motion of the individual robot components considering user space (Cartesian coordinate system) and robot control space (joint (drive) coordinate system).

52.4.1 Structure of VR Model

The individual 3D robot components (i.e. base and links of the robot) are exported separately from NX in the `*.wrl` format. Then, other parts of the VR world such as scene and background can be created in different graphical tools and stored in different graphical formats, e.g. in the `*.jpg` format. All the parts and components are consecutively in-lined to one VR model. Specifically, the robot components are gradually encapsulated one within the other in compliance with their order in robot structure.

The structure of the VR model that corresponds to this way is listed onwards in Table 52.2. Initial elements describe user viewpoints and the specific world (VR scene)—they are on the same level. Other elements (i.e. robot components) labelled ***(Transform)*** are encapsulated one (child, slave) within the other (parent, master) as already mentioned above.

52.4.2 Direct Kinematics

The VR structure closely corresponds to the direct kinematics, which uses three basic motions and three basic rotations in transformation matrices as follows:

Table 52.2 Listing of structure of VR model

```
ROOT
  SIDEview (Viewpoint)
    fieldOfView (SFFloat): real
    orientation (SFRotation): x y z ϱ
    position (SFVec3f): x y z
    description (SFString): SIDEview
  FRONTview (Viewpoint)
    ...
  TOPview (Viewpoint)
    ...
  World (Background)
    groundAngle/skyAngle (MFFloat)
    groundColor/skyColor (MFColor):rgb
    backUrl/bottomUrl/frontUrl/leftUrl/
    rightUrl/topUrl (MFString):[face.jpg]
  ...
  ...
  ...
  TrBASE (Transform)
    rotation (SFRotation): 0 0 0 0
    translation (SFVec3f): 0 0 0
    children (MFNode)
      BASE (Inline) url: [BASE.wrl]
      TrLINK1 (Transform)
        rotation (SFRotation): 0 0 1 q1
        translation (SFVec3f): 0 0 h0
        children (MFNode)
          LINK1 (Inline) url: [LINK1.wrl]
          TrLINK2 (Transform)
            rotation (SFRotation): 0 1 0 q2
            translation (SFVec3f): r1 0 h1
            children (MFNode)
              LINK2 (Inline) url: [LINK2.wrl]
              TrLINK3 (Transform)
                rotation (SFRotation): 0 1 0 q3
                translation (SFVec3f): 0 0 l2
                children (MFNode)
                  LINK3 (Inline) url: [LINK3.wrl]
                  TrLINK4 (Transform)
                    rotation (SFRotation): 1 0 0 q4
                    translation (SFVec3f): l3 0 0
                    children (MFNode)
                      LINK4 (Inline) url: [LINK4.wrl]
                      TrLINK5 (Transform)
                        rotation (SFRotation): 0 1 0 q5
                        translation (SFVec3f): l4 0 0
                        children (MFNode)
                          LINK5 (Inline) url: [LINK5.wrl]
                          TrLINK6 (Transform)
                            rotation (SFRotation): 1 0 0 q6
                            translation (SFVec3f): l5 0 0
                            children (MFNode)
                              LINK6 (Inline) url: [LINK6.wrl]
```

$$\begin{bmatrix} cos\, q_i & -cos\, \alpha_i\, sin\, q_i & sin\, \alpha_i\, sin\, q_i & a_i\, cos\, q_i \\ sin\, q_i & cos\, \alpha_i\, cos\, q_i & -sin\, \alpha_i\, cos\, q_i & a_i\, sin\, q_i \\ 0 & sin\, \alpha_i & cos\, \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{52.1}$$

The number of variable rotations q_i is six according to number of degrees of freedom (DOF) of the considered robot. Other indicated constants h_0, r_1, h_1, ℓ_2, ℓ_3, ℓ_4, ℓ_5, and ℓ_6 are constants of length-geometric robot parameters such as vertical distances, axis offsets, and lengths of robot arms.

52.4.3 Inverse Kinematics

Inverse Kinematics is necessary since input into the VR model is in rotational angles of individual robot joints. However, robot motion is given by Cartesian coordinates of its end effector considering user space. In this section, the principle of inverse kinematics will only be outlined due to its many routine solutions in robotics. If the robot IRB 140 [1] is considered, it is necessary to take into account that it has 6 DOF. They represent three Cartesian coordinates and three orientation angles. This situation is shown in Fig. 52.4.

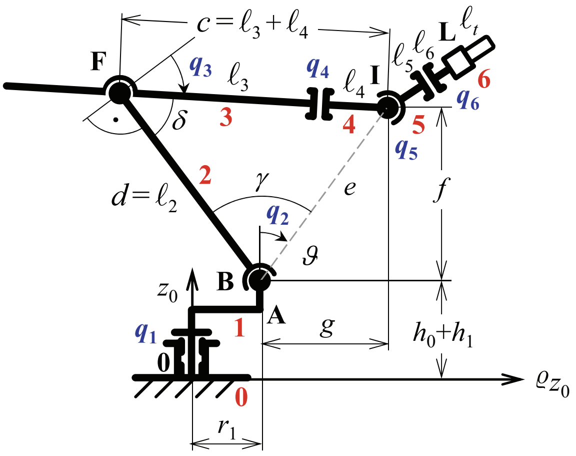
From a mathematical point of view, point **I** (axis A_5), or its Cartesian coordinates, has to be determined. It is possible due to local inverse kinematics for robot end effector point using orientation angles and lengths of Link5,6 + tool length, i.e. $\ell_5 + \ell_6 + \ell_t$, as in [5]. If the coordinates of point **I** are determined, then there is one triangle determined by vertices $\mathbf{B} = A_2$, $\mathbf{F} = A_3$, and $\mathbf{I} = A_5$. This triangle enables us to compute angles q_2 and q_3 in joints A_2 and A_3.

$$\begin{aligned} &g = \sqrt{I_x^2 + I_y^2} - r_1, \quad f = I_z - h_0 - h_1, \quad e = \sqrt{f^2 + g^2} \\ &\vartheta = \text{atan2}(f,\ g), \quad \delta = \arccos \frac{c^2 + d^2 - e^2}{2\, d\, c}, \quad \gamma = \arccos \frac{d^2 + e^2 - c^2}{2\, d\, e} \\ &q_2 = -(\vartheta + \gamma - \frac{\pi}{2}), \quad q_3 = \frac{\pi}{2} - \delta. \end{aligned} \tag{52.2}$$

Then, angle q_1 in joint $\mathbf{0} = A_1 = [0,\ 0,\ 0]^T$ can be determined from the robot top view by means of function $atan2$ ($arcus\ tangent$) of y and x coordinates of point $\mathbf{I} = A_5 = [I_x,\ I_y,\ I_z]^T$ towards the robot origin:

$$q_1 = \text{atan2}(I_y,\ I_x). \tag{52.3}$$

It is possible, since points **0** and **I** belong to the same vertical plane as well as longitudinal central axes of Link3 and Link4. The angles q_4 and q_6 are closely related and they are determined from angle q_5 obtained from three input orientation angles



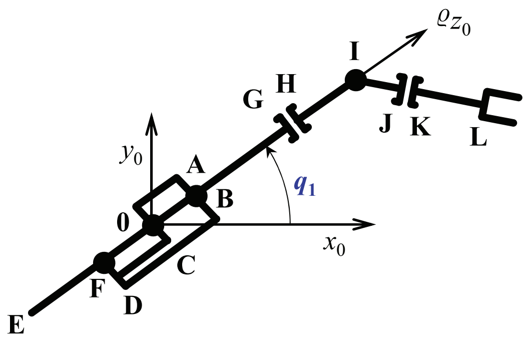


Fig. 52.4 2D views of robot structure for inverse kinematics

or respective points of robot wrist (Link4–6). Hence, obtained angles q_{1-6} are inputs for MPC design as well as for the 3D Visualization.

52.5 Simulation of Robot MPC

Model predictive control (MPC) belongs among the most popular perspective approaches to model-based control considered for application in industrial practice [3, 6]. For the purpose of the paper, let us consider one of the usual implementation ways. In robot motion control, MPC should generate executive control actions with respect to required motion trajectories. They are usually represented by ordered set of time and coordinates [4]. This way is considered in this paper.

52.5.1 Model of Robot Dynamics

The used MPC design exploits the following discrete state-space model:

$$\begin{aligned} \hat{x}_{k+1} &= A_k\, x_k + B_k\, u_k \\ y_k &= C x_k. \end{aligned} \tag{52.4}$$

The model (52.4) is obtained from a nonlinear differential equation of second order $\ddot{y} = f(y, \dot{y}) + g(y)u,\ \ y \equiv q$ that is given by Lagrange equations of second type. The nonlinear model is decomposed into a linear-like state-space model and then discretized [6]. The Lagrange equations of second type are defined as follows:

$$\frac{d}{dt}\left(\frac{\partial E_k}{\partial \dot{q}}\right)^T - \left(\frac{\partial E_k}{\partial q}\right)^T + \left(\frac{\partial E_p}{\partial q}\right)^T = \ \tau, \tag{52.5}$$

where $q, \dot{q}, E_k, E_p$, and τ are generalized coordinates and their appropriate derivatives, total kinetic and potential energy, and vector of generalized force effects associated with generalized coordinates [11].

Thus, the model (52.4) or its matrices A_k and B_k, respectively, are updated with respect to a measured state and nonlinear robot dynamics in every time-sampling instant before new computation of the vector of control actions u_k.

52.5.2 MPC Design

MPC design consists in a minimization of the criterion with appropriate cost function [9]:

$$\min_{U_k}\ J_k\,(\hat{Y}_{k+1}, U_k, W_k) \tag{52.6}$$

$$\begin{aligned} \text{subject to: } \hat{x}_{k+i} &= \quad A\, x_{k+i-1} + B\, u_{k+i-1} \\ \hat{y}_{k+i} &= \quad C\hat{x}_{k+i} \quad \forall\, i = 1, \cdots, N\ , \end{aligned}$$

where $A = A_{k+i-1} = A_k,\ B = B_{k+i-1} = B_k,\ \forall\, i = 1, \cdots, N$ from (52.4) are updated for every minimization in discrete instants k, and N is a prediction horizon. The cost function J_k itself is defined as follows:

$$\begin{aligned} J_k &= \sum_{i=1}^{N}\{||Q_y\,(\hat{y}_{k+i} - w_{k+i})||_2^2) + ||Q_u\, u_{k+i-1}||_2^2\} \\ &= (\hat{Y}_{k+1} - W_{k+1})^T\, Q_Y^T\, Q_Y\,(\hat{Y}_{k+1} - W_{k+1}) \\ &\quad + U_k^T\, Q_U^T\, Q_U\, U_k, \end{aligned} \tag{52.7}$$

where U_k, $\hat{Y}_{k+1}$ and W_{k+1} stand for the sequences of control actions, output predictions, and references, respectively,

$$U_k \;\; = [u_k^T, \cdots, u_{k+N-1}^T]^T \tag{52.8}$$

$$\hat{Y}_{k+1} = [y_{k+1}^T, \cdots, y_{k+N}^T]^T \tag{52.9}$$

$$W_{k+1} = [w_{k+1}^T, \cdots, w_{k+N}^T]^T . \tag{52.10}$$

Furthermore, Q_Y and Q_U are matrices of penalization that are defined as follows:

$$Q_\diamond^T Q_\diamond = \begin{bmatrix} Q_*^T Q_* & & 0 \\ & \ddots & \\ 0 & & Q_*^T Q_* \end{bmatrix}, \tag{52.11}$$

where the symbolic subscripts $\diamond$, $*$ have the following interpretation: $\diamond \in \{YW,\ U\}$ and $* \in \{yw,\ u\}$.

The equations of predictions express functional estimates of the output elements in $\hat{Y}_{k+1}$ relative to searched vector of control actions U_k within the prediction horizon N

$$F_k = \left.\begin{bmatrix} CA \\ \vdots \\ CA^N \end{bmatrix}\right|_k , \quad G_k = \left.\begin{bmatrix} CB & \cdots & 0 \\ \vdots & \ddots & \vdots \\ CA^{N-1}B & \cdots & CB \end{bmatrix}\right|_k . \tag{52.12}$$

Then for the nonzero matrix Q_U, the predictive control law can be written as follows:

$$\begin{aligned} u_k = MU_k = M(G_{p,k}^T\, Q_Y^T\, Q_Y\, G_{p,k} + Q_U^T\, Q_U)^{-1} \\ \times G_{p,k}^T\, Q_Y^T\, Q_Y\, (R_{k+1} - F_{p,k}\, x_k)\,, \end{aligned} \tag{52.13}$$

where a rectangular matrix M is defined as follows:

$$M = \left[\, I_{m_u},\ 0_{m_u},\ \cdots,\ 0_{m_u} \,\right]. \tag{52.14}$$

Thus, the matrix M selects only the appropriate control actions corresponding to the time instant k. The obtained vector of control actions represent reference torques expected in individual robot joints A_1–A_6.

52.5.3 Simulation and 3D Visualization

A specific simulation can be done in MATLAB only or in MATLAB/Simulink. To construct a VR world, it is useful to use 3D World Editor that can be reached from the Simulink model, e.g. in the model shown in Fig. 52.5.

The mentioned model contains a block of input from the workspace, several routing signals blocks, and block VR Sink (Simulink 3D Animation Toolbox [8]. It is the biggest block concurrently masked by robot figure). It serves for connection to the VR world. This is not the only way.

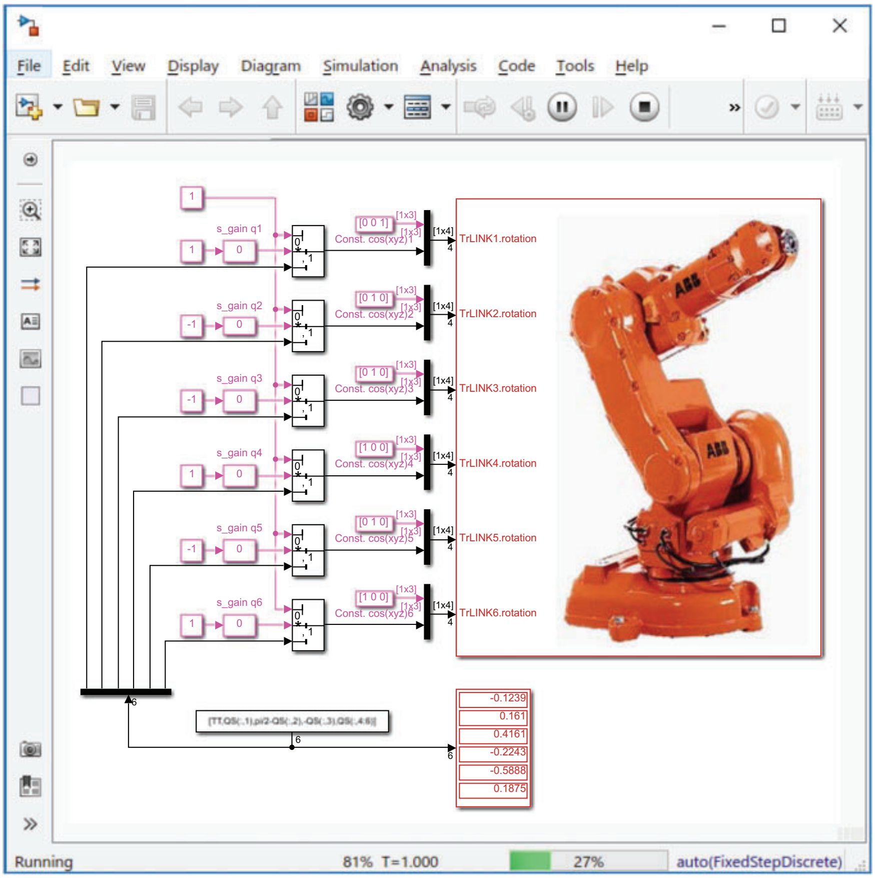


Fig. 52.5 Simulink model with connection to VR world

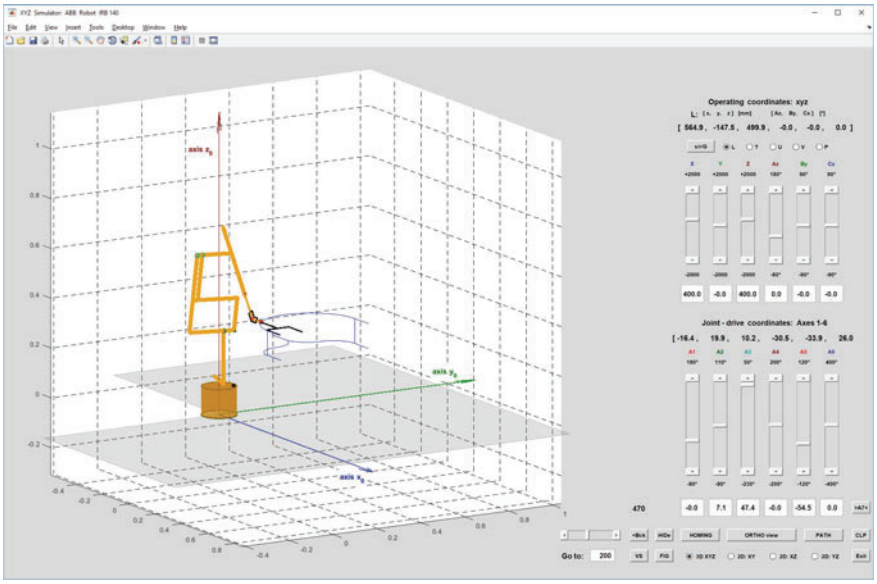


Fig. 52.6 Simulator of ABB robot IRB 140 with 3D wire-frame model in MATLAB figure

Another way is to use commands from MATLAB, connecting through the mentioned 3D Animation Toolbox to VR world, such as

```
myworld = vrworld('*.wrl');
open(myworld);
view(myworld);
⋮
myworld.TrLINKi.rotation = [0 0 1 qi]; and others.
```

In the following figures, Figs. 52.6 and 52.7, there is an illustration of the realization of the visualization. Figure 52.6 shows a simple 3D wire robot model constructed by simple MATLAB elements such as lines and surfaces, whereas Fig. 52.7 shows a fully spatial, 3D robot model consisted of individual volume parts such as robot base (basic frame) and robot links including simple scene.

Last but not the least figure, Fig. 52.8 demonstrates results of motion control using MPC applied to the ABB robot IRB 140 along the considered trajectory specified in Table 52.1. There are necessary kinematic quantities such as position, velocity, and acceleration profiles in the Cartesian coordinate space and corresponding joint angles in the joint space (top to bottom up to the fifth row). The last, the sixth row, shows individual control actions—torques applied to robot joints.

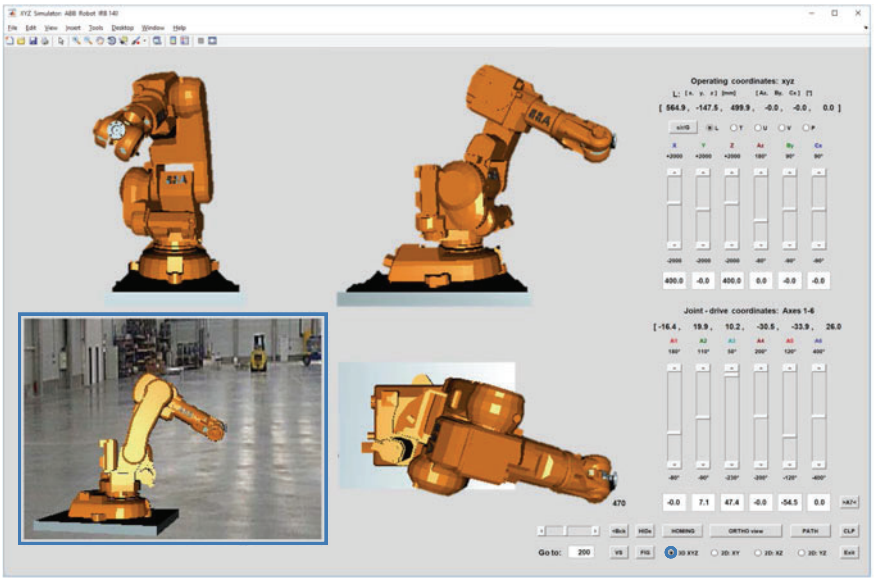


Fig. 52.7 Simulator of ABB robot IRB 140 with 3D volume model in MATLAB figure using 3D Animation Toolbox

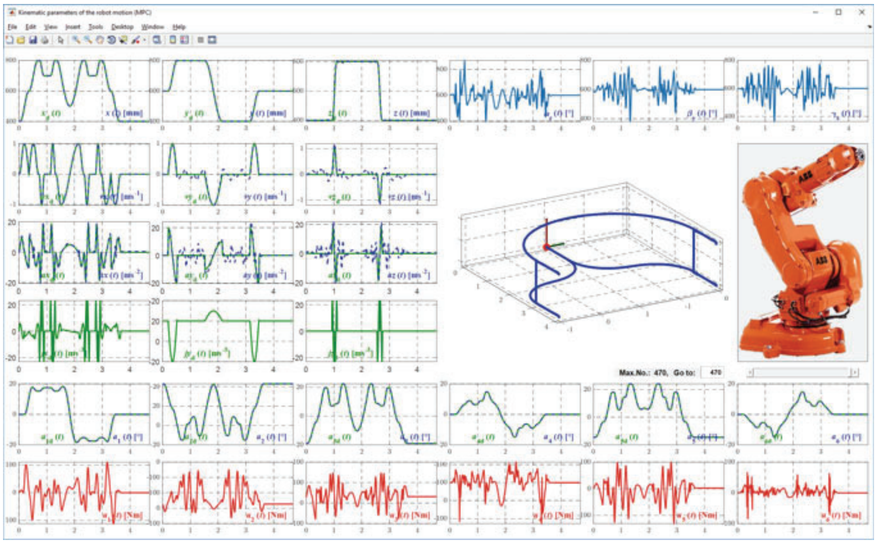


Fig. 52.8 Demonstration of results of MPC motion control of ABB robot IRB 140

52.6 Conclusion

The paper introduces a promising way of linking 3D visualization with MPC design and simulation for industrial robot control. Both used tools, NX software and MATLAB/Simulink, are specialized for specific work. However, the interconnection of the visualizing tool with the mathematical-computing environment is under continuous effort. The proposed way indicates one possible realization of a full physically modelled virtual digital factory that can offer online full digital image of the reality including motion dynamics of robots.

In future work, we would like to focus on scene extension and collaboration of more robotic systems to simulate not only their kinematics (usually offered) but also a fully compliant dynamics and model-based motion control. The goal is to construct a realistic digital factory that is able to demonstrate realistic actions and motions of the industrial robotic systems.

References

1. ABB: ABB robot IRB140 - product manual & CAD data. online (2019), cited 2019-05-26
2. Axiom Tech: NX CAD. online (2019), cited 2019-05-26
3. Belda, K.: Nonlinear design of model predictive control adapted for industrial articulated robots. In: Proc. of the 15th Int. Conf. on Informatics in Control, Automation and Robotics. pp. 71–80. INSTICC, Scitepress. (2018)
4. Belda, K., Novotný, P.: Path simulator for machine tools and robots. In: Proc. of the 17th Int. Conf. on Methods and Models in Automation and Robotics. pp. 373–378. West Pomeranian University of Technology (2012)
5. Belda, K., Rovný, O.: Predictive control of 5 DOF robot arm of autonomous mobile robotic system motion control employing mathematical model of the robot arm dynamics. In: Proc. of the 21th Int. Conf. on Process Control. pp. 339–344. Slovak University of Technology in Bratislava (2017)
6. Belda, K., Záda, V.: Predictive control for offset-free motion of industrial articulated robots. In: Proc. of the 22nd IEEE Int. Conf. on Methods and Models in Autom. and Robotics. pp. 688–693. West Pomeranian Univ. of Technology (2017)
7. Dvořák, K.: Management of parametric CAD model by external tools. Mechanical and Aerospace Engineering IV **390**(11), 616–620 (2013)
8. MathWorks: Simulink 3D animation - user's guide. online (2019), cited 2019-05-26
9. Ordis, A., Clarke, D.: A state-space description for gpc controllers. J. Systems SCI. **24**(9), 1727–1744 (1993)
10. Ricci, F., Bedolla, J., Gomez, J., Ciabert, P.: PMI: a PLM approach for the management of gemetrical and dimensional controls in modern industries. Computer-Aided Design and Applications **11**(1), 36–43 (2014)
11. Siciliano, B., et al., L.S.: Robotics: Modelling, planning and control. Springer (2009)
12. Siemens: NX docomentation. online (2019), cited 2019-05-26
13. Zohaib, A., Asad, A.: Modeling and simulation of a lower-body wearable exoskeleton using robotics techniques. Int. J. of Mechanical Engineering and Robotics Research 7(3), 313–318 (2018)