KUKA Sunrise Toolbox: Interfacing Collaborative Robots with MATLAB

Mohammad Safeea and Pedro Neto

***Abstract*—Collaborative robots are increasingly present in our lives. The KUKA LBR iiwa equipped with the KUKA Sunrise.OS controller is a representative example of a collaborative/sensitive robot. This tutorial presents a MATLAB Toolbox, the KUKA Sunrise Toolbox (KST), to interface with KUKA Sunrise.OS using MATLAB. The KST contains functionalities for networking, soft real-time control, point-to-point motion, setters/getters of parameters, general purpose, and physical interaction. KST includes around 100 functions and runs on a remote computer connected with the KUKA Sunrise controller via Transmission Control Protocol/Internet Protocol (TCP/IP). The KST potentialities are demonstrated in nine application examples.**

***Index Terms*—KUKA Sunrise Toolbox, MATLAB, KUKA LBR iiwa, KUKA Sunrise.OS, collaborative robotics.**

I. Introduction

A. Motivation and Related Work

COLLABORATIVE robots have been extensively studied, becoming increasingly safe, intuitive to use, and robust. The KUKA LBR iiwa equipped with the KUKA Sunrise.OS controller is a representative example of a collaborative/sensitive robot with applications in several areas. KUKA Sunrise.OS equips the KUKA LBR iiwa and the mobile platform KMR iiwa. However, due to its functionalities (advanced programming, planning and configuration), it is expected that the Sunrise.OS will also be available for other KUKA robots in the future. These robots are being used in several projects for advanced applications, such as ColRobot, EURECA and CogIMon projects [1], [2], with possible implementation in various domains as demonstrated in the KUKA innovation awards. In these applications, the robot works side-by-side with the co-worker, providing assistance in an intuitive and safe manner.

In the robotics field, several MATLAB toolboxes have been introduced. One of the most popular is the Robotics Toolbox for MATLAB [3]. This toolbox includes functionalities for robotic manipulators, such as homogeneous transformations, forward and inverse kinematics, forward and inverse dynamics, and trajectory generation. The Dynamics Simulation Toolbox for industrial robot manipulators can be used for simulating robot dynamics in addition to other functionalities [4]. The DAMAROB Toolbox allows kinematic and dynamic modeling of manipulators [5]. The KUKA control toolbox is dedicated to the motion control of KUKA manipulators equipped with the KRC controller [6]. JOpenShowVar is a Java-based open-source platform to interface KUKA industrial robots equipped with the controller KR C4 and KR C2 [7].

Recently, KUKA launched the LBR iiwa series of manipulators [8], a commercial version of the KUKA-DLR lightweight robot [9]. The LBR iiwa is a lightweight sensitive robot with seven axes (redundant). Each axis contains multiple sensors allowing position and impedance control. These robots are programmed using the KUKA Sunrise.Workbench (the programming environment for KUKA Sunrise). IO connectors and an EtherCAT interface are available in the robot flange.

From an external computer, the user can interface with Sunrise.OS using the Fast Robot Interface (FRI) [9] and Robot Operating System (ROS) [10]. The FRI is a platform for controlling the KUKA iiwa remotely from a personal computer (PC) allowing hard real-time control at rates of up to 1 kHz. ROS is popular among the research community, allowing users to use an external PC to interface with robotic systems. The iiwa_stack [11] is one of the most commonly used packages for interfacing Sunrise.OS with ROS. This package is built upon the Smart.Servo interface only, as such it provides a soft real-time control capability, allowing the user to control iiwa from ROS on-the-fly. On the other hand, Table I, KST is the unique MATLAB-based interface for Sunrise.OS. It is a plug-and-play solution that does not require a special configuration in the controller and covers a wide range of Sunrise.OS features, especially for human-robot interaction capabilities. As compared to the iiwa_stack, the KST also includes extra functionalities for hand-guiding, precision hand-guiding, non-blocking motion calls, conditional motion calls, and point-to-point motion calls (arcs, lines, ellipses) among others. In addition, KST is provided with numerous examples ranging from simple tutorials on the straightforward implementation of its functions to more complex examples that involve robot control based on external hardware/sensor inputs. The KST can be used in education, research and industry. Owing to its MATLAB interface, KST allows to speed up the development of applications for KUKA Sunrise.OS. Moreover, it makes the development of robot applications accessible to people with basic skills in MATLAB, even if they are not particularly experts in programming.

The KUKA Sunrise.OS controller is programmed using Java, allowing the implementation of complex algorithms in the robot controller. The use of an external computer to interface with the robot has a negative impact on the real-time execution of robot's commands due to the communication delay. In the KST, tests revealed a three to four millisecond communication delay when sending and receiving commands

Mohammad Safeea is with the Department of Mechanical Engineering, University of Coimbra, Portugal and with Arts et Métiers, ParisTech, France, e-mail: ms@uc.pt.

Pedro Neto is with the Department of Mechanical Engineering, University of Coimbra, Coimbra, Portugal, e-mail: pedro.neto@dem.uc.pt.

TABLE I
IIWA_STACK (ROS) AND KST INTERFACES

	iiwa_stack	KST
Operating system	Linux	Linux, Windows, Mac
Programming languages	C++ or Python	MATLAB
Required knowledge	ROS	MATLAB
3D simulator	gazebo	v-rep

to and from the robot, as detailed in the **extended version of this tutorial in the multimedia material**. In this scenario, the use of an external computer is advantageous in several cases:

1) To interface with multiple external devices;
2) Easy integration of external software modules and hardware devices;
3) Implementation of complex algorithms (image processing, machine learning, etc.) requiring high computational power;
4) When the amount of computations involved is relatively high so that performance is limited by the robot controller hardware. Unlike the robot controller, the computer hardware can be easily upgraded.

B. Original Contribution

The KUKA Sunrise Toolbox (KST) contains more than 100 functions and presents multiple advantages:

1) Easy and fast interaction with the robot from an external computer running the KST. It is an open source solution provided under MIT license;
2) KST integrates diverse functionalities including precision robot hand-guiding[1], kinematics, motion definition, among others;
3) Speed up the development of advanced robot applications in MATLAB. Complex algorithms and advanced mathematical tools supported by MATLAB (for example matrix operations and data filtering) can be implemented in an external computer. Existing software modules/toolboxes (vision, machine learning, statistics, etc.) can also be integrated, allowing to extend the applications with new functionalities;
4) The KST makes the KUKA LBR iiwa manipulators accessible to a wide variety of people from different backgrounds, and opens the door of collaborative robotics to many potential new users for academic, educational and industrial applications. MATLAB is widely used to accelerate the process of research implementation.
5) KST runs inside MATLAB, so it can be used on different operating systems: Windows, Linux and macOS.

II. KUKA SUNRISE TOOLBOX

The KST functions are divided into seven categories:

1) Networking – Used to establish (and terminate) connection with the robot controller;

[1]Precision hand-guiding refers to robot hand-guiding at the end-effector level (translations and rotations for precision positioning) [12], which is different from KUKA's off-the-shelf hand-guiding at the joint level.

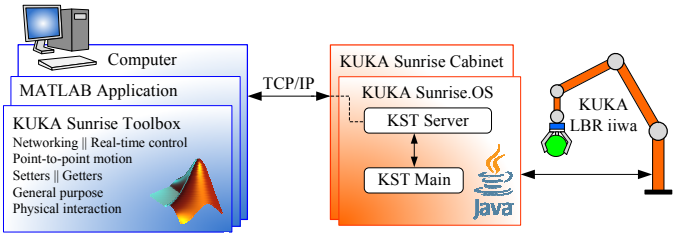

Fig. 1. Architecture and communication scheme of the KST. The KUKA Sunrise cabinet is the physical robot controller.

2) Soft real-time control – Used to activate/deactivate soft real-time functionalities for motion and impedance control. In such a case, robot's path can be updated online from external sensors data with simultaneous control-feedback capability. These functionalities are built upon the DirectServo and the SmartServo from KUKA;
3) Point-to-point motion – Used to perform point-to-point motion in joint space and in Cartesian space. This category also includes non-blocking and conditional/interruptible motion functions;
4) Setters – Used to set parameter values in the robot controller (robot poses, LED and IO connectors);
5) Getters – Used to get parameter values from the robot controller (joint angles, end-effector position, end-effector orientation, force/moment acting on the end-effector, joints torques, IO connectors);
6) General purpose – Used to calculate forward and inverse kinematics, mass matrix, Coriolis matrix and Jacobian;
7) Physical interaction – Used to activate/deactivate hand-guiding, precision hand-guiding [12] and double tap detection.

The KST runs on an external/remote computer and communicates with KUKA Sunrise via TCP/IP through an Ethernet network using the X66 connector of the robot, Figure 1. The KST implements a TCP/IP client which communicates with the Java server (KST Server and KST Main) running on the Sunrise.OS. Both KUKA iiwa manipulators (KUKA iiwa 7 R800 and KUKA iiwa 14 R820) are supported.

The main functions of KST are detailed and illustrated with implementation examples in the extended version of this tutorial in the multimedia material. The KST toolbox can be freely downloaded from the GitHub repository: https://github.com/Modi1987/KST-Kuka-Sunrise-Toolbox provided under MIT license.

III. APPLICATION EXAMPLES

The robot, with a pen mounted on the flange, produces the drawing of a circle on the top of a white box. This task is achieved using the KST point-to-point functionalities that support arc motion. An illustrative example/algorithm is given in Figure 2.

This task can also be achieved using the inverse kinematics solver of the toolbox and the soft real-time control functionalities of the KST. The end-effector circular path is calculated while the the robot is moving. The inverse kinematics solver is used to calculate the joint angles of

Algorithm 1: Drawing a circle (MATLAB code)

```
% Instantiate the KST object
ip = '172.31.1.147'; % IP of the robot controller
arg1 = KST.LBR7R800; % Robot type/model
arg2 = KST.Medien_Flansch_elektrisch; % Flange type/model
Tef_flange = eye(4); % End-effector to flange transform
iiwa=KST(ip,arg1,arg2,Tef_flange);
% Connect to the robot controller
iiwa.net_establishConnection();
% Define circle radius and robot velocity
r = 50; vel = 150;
% Define the center of the circle as the current end-
effector position
Cen = iiwa.getEEFPos();
% Define sPoint as the starting point of the circle
sPoint = Cen; sPoint{1} = sPoint{1}+r;
% Move the end-effector to the starting point of the circle
iiwa.movePTPLineEEF(sPoint,vel);
% Specify the parameters of the arc, namely the angle theta
% subtended by the arc at the center of the rotation and the
% XY coordinate of the center of the arc
theta = -2*pi;
c = [Cen{1}; Cen{2}];
% Move the end-effector to perform the arc motion
iiwa.movePTPArcXY_AC(theta,c,vel);
```

Fig. 2. Drawing a circle algorithm in MATLAB code.

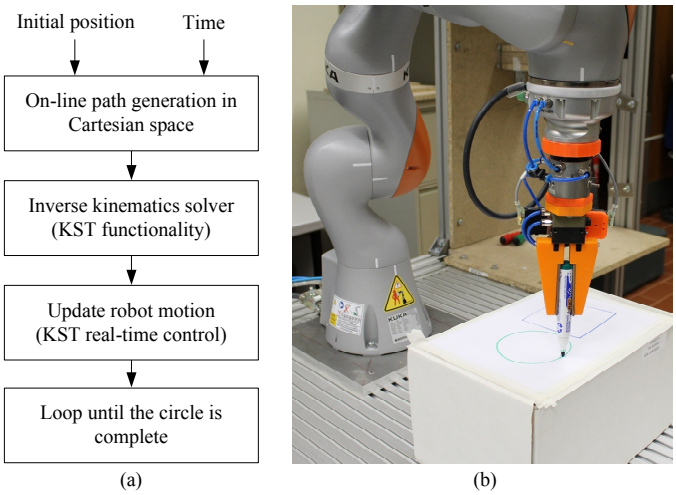

Fig. 3. Example 1: (a) flow chart of MATLAB script and (b) robot drawing the circle.

the robot from the circular path. The calculated angles are relayed to the Sunrise controller through the KST such that the robot end-effector executes the circular path. Figure 3 (a) shows the flow chart of the control scheme and Figure 3 (b) shows a snapshot of the robot drawing a circle. Since robot motion is being generated online using the soft real-time control functionalities of the KST, it is possible to adjust the path on-the-fly, to change the velocity or to stop the motion if desired. The example code is in the MATLAB file `KSTclass_Tutorial_drawCircle.m`.

The KST was also used to implement a practical use case related to the assembly of two parts joined by screws in which the human co-worker and the robot share workspace and sub-tasks. As such, KST was used to implement a human-robot collision avoidance system based on the well-known potential fields method [13]. The screwing operation, Figure 4, of a single screw was divided into 3 sub-tasks:

1) The human co-worker approaches the workpiece to place the screw into the hole while the robot moves away to avoid collision. In this phase the co-worker rotates the screw (1-2 turns) to fix it and leaves the area. The collision avoidance system is able to adjust the off-line pre-planned paths smoothly and on-the-fly for avoiding collisions with the dynamic co-worker, Figure 4 (a-c);
2) The robot automatically returns to the pre-planned path to tighten the screw. It approaches the screw head from top, while the tool attached to the robot end-effector starts rotating. When a given torque is reached the tool stops rotating and the robot moves up. In this phase, when the robot reaches below a predefined distance above the screw head the collision avoidance is deactivated. In this scenario the robot velocity is relatively reduced, limiting the risk for the human co-worker, Figure 4 (d-f);
3) Finally, the human approaches the workpiece to apply a final manual tightening with adequate pressure. The robot moves away to avoid collision, Figure 4 (g-h).

In this system, a magnetic tracker is used to capture the pose of the human around the robot. Each magnetic tag provides position and orientation data, which are used as inputs to compute the minimum distance between the human and the robot, both geometrically approximated by capsules. The robot motion was controlled using the soft real-time control functions provided by the KST. We conducted a quantitative analysis by recording the human-robot minimum distance, robot velocity and the robot joint angles, Figure 5. This analysis was focused in the first sub-task and in the beginning of second sub-task, Figure 4 (a-e). At the beginning the robot is stationary. When the human co-worker approaches the workpiece the human-robot minimum distance decreases to a minimum of 0.3 meters and the robot reacts to avoid collision, Figure 5. After this process, the co-worker is placing the screw, the minimum distance is stable and the robot is stopped keeping a given safe distance. When the human moves away (second sub-task) the robot returns back to the workpiece. The example video as well as another collision avoidance example with extra detail is in the extended version of this tutorial in the multimedia material.

Nine application examples on a KUKA iiwa 7 R800 manipulator demonstrate the performance and ease of use of KST for drawing geometries, DirectServo control, human-robot collision avoidance, teleoperation, hand-guiding and teaching, interfacing Sunrise with v-rep, and controlling iiwa using Graphical User Interface. The application examples also include two practical use cases one for assembly operation using screws and the other for pick-and-place operation. **These examples are detailed in the extended version of this tutorial in the multimedia material**.

IV. Conclusion

According to users' feedback (students, researchers and industry engineers), the proposed toolbox is a useful and intuitive tool to interface with KUKA Sunrise.OS, namely to speed

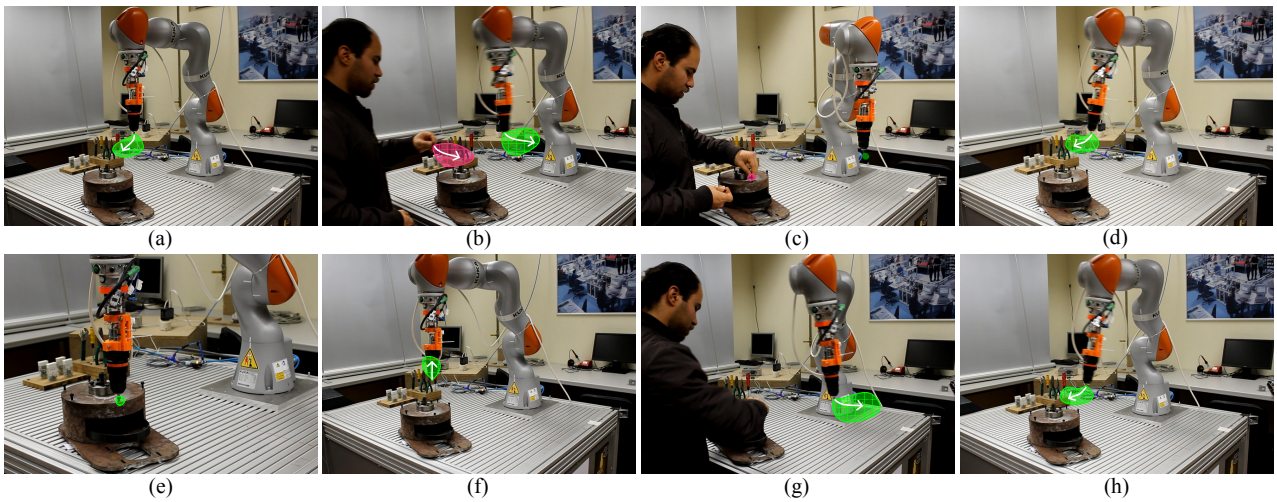

Fig. 4. The robot smoothly avoids collision when the human co-worker approaches to perform the screwing operation.

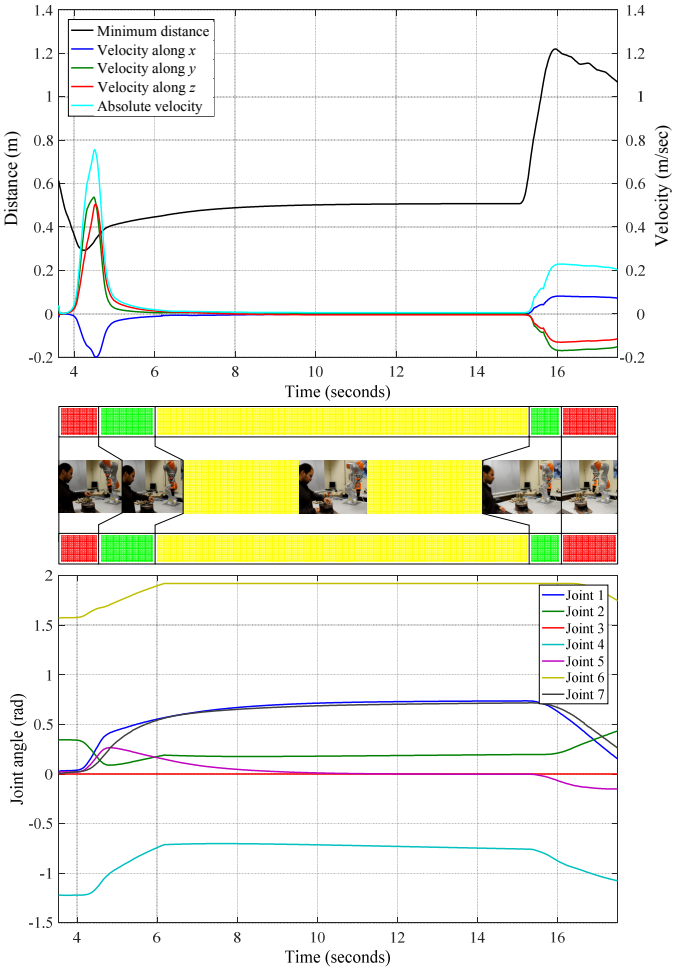

Fig. 5. Human-robot minimum distance, robot velocity and joint angles recorded during experimental tests.

up the development and implementation of robot applications. The KST functionalities demonstrated to be advantageous in the implementation of advanced robot applications. KST also facilitates the integration of external hardware, data processing and the implementation of complex algorithms using existing toolboxes.

V. Acknowledgment

This research was partially supported by Portugal 2020 project DM4Manufacturing POCI-01-0145-FEDER-016418 by UE/FEDER through the program COMPETE2020, and the Portuguese Foundation for Science and Technology (FCT) SFRH/BD/131091/2017.

References

[1] L. Roveda, S. Haghshenas, A. Prini, T. Dinon, N. Pedrocchi, F. Braghin, and L. M. Tosatti, "Fuzzy impedance control for enhancing capabilities of humans in onerous tasks execution," in *2018 15th International Conference on Ubiquitous Robots (UR)*. IEEE, 2018, pp. 406–411.

[2] S. S. M. Salehian and A. Billard, "A dynamical-system-based approach for controlling robotic manipulators during noncontact/contact transitions," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2738–2745, Oct 2018.

[3] P. I. Corke, "A robotics toolbox for matlab," *IEEE Robotics Automation Magazine*, vol. 3, no. 1, pp. 24–32, Mar 1996.

[4] M. Toz and S. Kucuk, "Dynamics simulation toolbox for industrial robot manipulators," *Computer Applications in Engineering Education*, vol. 18, no. 2, pp. 319–330, 2010. [Online]. Available: http://dx.doi.org/10.1002/cae.20262

[5] M. Bellicoso, *DAMAROB Toolbox [On-line]*. Available: http://www.damarob.altervista.org.

[6] F. Chinello, S. Scheggi, F. Morbidi, and D. Prattichizzo, "Kuka control toolbox," *IEEE Robotics Automation Magazine*, vol. 18, no. 4, pp. 69–79, Dec 2011.

[7] F. Sanfilippo, L. I. Hatledal, H. Zhang, M. Fago, and K. Y. Pettersen, "Controlling kuka industrial robots: Flexible communication interface jopenshowvar," *IEEE Robotics Automation Magazine*, vol. 22, no. 4, pp. 96–109, Dec 2015.

[8] *KUKA LBR iiwa series [On-line]*. Available: https://www.kuka.com/en-my/products/robotics-systems/industrial-robots/lbr-iiwa.

[9] R. Bischoff, J. Kurth, G. Schreiber, R. Koeppe, A. Albu-Schaeffer, A. Beyer, O. Eiberger, S. Haddadin, A. Stemmer, G. Grunwald, and G. Hirzinger, "The kuka-dlr lightweight robot arm - a new reference platform for robotics research and manufacturing," in *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*, June 2010, pp. 1–8.

[10] *ROS Industrial support for the KUKA LBR iiwa [On-line]*. Available: http://wiki.ros.org/.
[11] *ROS Indigo/Kinetic metapackage for the KUKA LBR IIWA R800/R820 [On-line]*. Available: http://github.com/IFL-CAMP/iiwa_stack/.
[12] M. Safeea, R. Bearee, and P. Neto, *End-Effector Precise Hand-Guiding for Collaborative Robots*. Advances in Intelligent Systems and Computing 694, Springer International Publishing, 2018, pp. 595–605. [Online]. Available: https://doi.org/10.1007/978-3-319-70836-2_49
[13] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.