Demo: FoReCo – a forecast-based recovery mechanism for real-time remote control of robotic manipulators

Milan Groshev
Universidad Carlos III de Madrid
Spain

Javier Sacido
Telcaria Ideas S.L.
Spain

Jorge Martín-Pérez
Universidad Carlos III de Madrid
Spain

ABSTRACT

This demonstration presents FoReCo [4], a solution to recover lost control commands in remotely controlled robots. In the demonstration, visitors use a joystick to remotely control a robotic arm under the presence of packet losses in the wireless medium. The lost control commands result in a distorted trajectory of the robotic arm, thus, we deploy FoReCo to recover lost control commands using an ML model that we train with a real-world dataset. The demonstration shows how FoReCo recovers the lost commands, and how the robot arm operates smoothly despite the losses that are present in the wireless medium.

CCS CONCEPTS

• **Networks** → *Cyber-physical networks.*

ACM Reference Format:
Milan Groshev, Javier Sacido, and Jorge Martín-Pérez. 2022. Demo: FoReCo – a forecast-based recovery mechanism for real-time remote control of robotic manipulators. In *ACM SIGCOMM 2022 Conference (SIGCOMM '22 Demos and Posters), August 22–26, 2022, Amsterdam, Netherlands.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3546037.3546047

1 INTRODUCTION

Real-time remote control of robotic manipulators is of high interest for industry 4.0 use cases [2], since it reduces costs and the exposure of operators to hazard situations. However, the presence of packet losses in the the network lead to trajectory deviations that prevent the robot from achieving the expected 99.9999% reliability [1].

The mechatronics and robotic community propose to overcome the lost/delayed control commands using time domain passivity-based solutions [3, 6, 11], or wave-variable passivity-based approaches [9, 10]. But the proposed approaches assume constant network delays [6], or delays with small variability [3, 6, 9–11]. Such assumptions on the network delay do not apply for robots connected using IEEE 802.11 wireless technology, which suffers from uncontrolled packet losses and delays. Specially in industrial scenarios with electromagnetic interference jamming the wireless channel.

SIGCOMM '22 Demos and Posters, August 22–26, 2022, Amsterdam, Netherlands

ACM ISBN 978-1-4503-9434-5/22/08.
https://doi.org/10.1145/3546037.3546047

Figure 1: FoReCo forecasts/infers control commands lost in the wireless communication between the remote controller and the robotic arm.

2 FORECO SOLUTION

Rather than making assumptions on the network delays, we propose FoReCo [4], a Forecast recovery mechanism for Remote Control that infers lost/delayed control commands. FoReCo feeds the robot drivers with the lost/delayed commands it recovers through forecasting – see Figure 1.

FoReCo receives the remote control commands and checks if they are received at the expected frequency, e.g., each 150ms. In case a command does not arrive on time (due to 802.11 collisions or interference not recovered with error-correction), FoReCo forecasts/infers the out of time command using the recent command history.

In this demonstration FoReCo uses a multinomial logistic regression that we train using a real-world dataset of an experienced robotic arm operator. The ML model receives as **input**: (1) the position of each joint $j_i(t), \forall i$; (2) for how long each joint has been moving without stopping $\max_{t_0<t} \{t - t_0 : \ j_i(\tau) > 0, \forall \tau \in [t_0, t]\}, \ \forall i$; and (3) the angular derivative of each joint $\frac{d}{d\alpha} j_i(t), \forall i$. Using the input (1)-(3), FoReCo runs the multinomial logistic regression and **outputs** if the lost command was an up/down movement, a righ/left sweep, or a grab/release action. The repetitive pick-and-place task fosters FoReCo's accuracy when it forecasts lost remote control commands.

3 FORECO DEPLOYMENT

In order to deploy FoReCo we create a Network Service (NS) with six Virtual Network Functions (VNFs) illustrated as red boxes in Figure 2: (1) the **FoReCo** VNF to assess the command recovery; (2) the robot **drivers** VNF to control and monitor the robotic arm hardware; (3) the **niryo-one-interface** and (4) **niryo-one-motion** to provide high-level abstraction for the core robot functionalities; (5) the **niryo-one-web** VNF to provide a set of tools (e.g., web interface, joystick/automated robot control, calibration) that facilitate the user interaction with the robot; and (6) the **niryo-ros-master** as a central entity to coordinate the communication among VNFs. The niryo-one-web/interface/motion/master and the drivers VNFs are

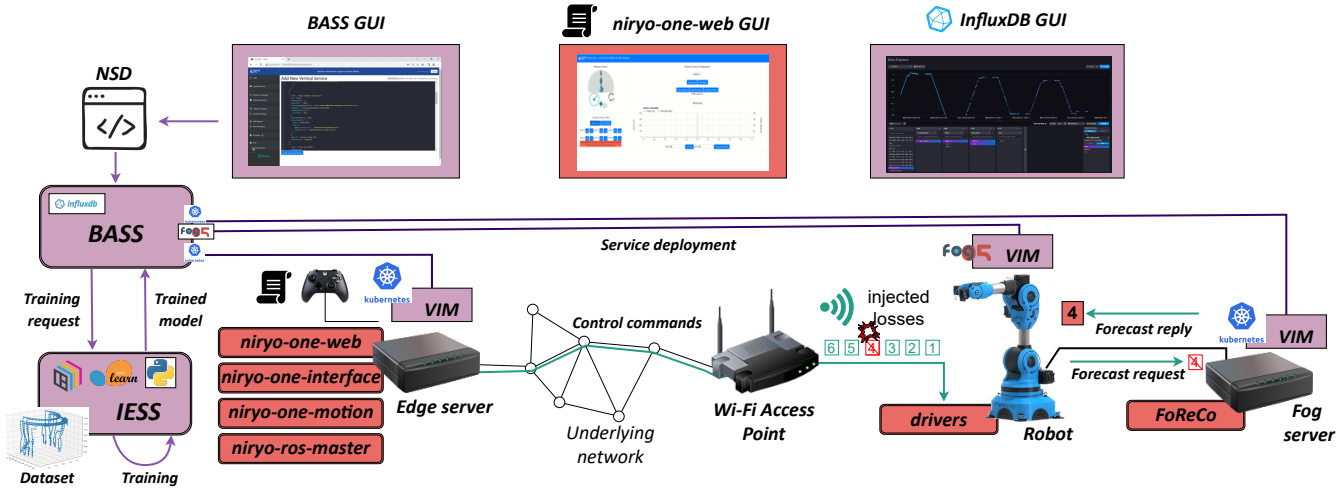

Figure 2: FoReCo-assisted remote control (red) deployed with DEEP modules (purple). Three GUIs are used to deploy the service (left), calibrate/control the robot (middle), and visualize FoReCo command recovery (right).

implemented using the Robot Operating System (ROS) [8], and the FoReCo VNF is implemented using scikit-learn [7] to forecast/infer lost commands. The resulting NS with the aforementioned VNFs is deployed on top of a joint fog05/Kubernetes cluster where the robot drivers VNF is deployed as a ROS native app using fog05. The niryo-one-web/interface/motion/master are deployed as containers using Kubernetes.

Once FoReCo is packed inside the aforementioned NS, we use the DEEP platform [5] for its automated ML training, deployment, life-cycle management, and termination. While the BASS module of the DEEP simplifies and automates the creation and management of the NS by being the logical entry point for the user, the IESS module facilitates the training and provisioning of FoReCo by providing an intent-driven approach and AutoAI solutions. The deployment through the DEEP platform proofs that FoReCo is ready to be integrated in real world platforms.

4 DEMONSTRATION OVERVIEW

As explained in Section 3, in this demonstration we deploy FoReCo with the DEEP platform. The demonstration shows the benefits of using FoReCo to perform forecast-based recovery of lost control commands in a Remote Control application. Moreover, we also do a live training of FoReCo using a real-world dataset of an experienced driver – see [4] for more details. The demonstration consists of four steps:

(1) First, we deploy the NS uploading its descriptor (NSD) in the BASS GUI. Then, the IESS trains FoReCo's multinomial model, and the BASS instantiates the NS as fog05/Kubernetes cluster that consists of: (*i*) FoReCo as a Kubernetes container in the Fog server.; (*ii*) the Drivers as a fog05 native application in the robot; and (*iii*) the nyrio-one-web/interface/motion and the ROS master as Kubernetes containers in the Edge server.
(2) Once DEEP finishes the NS deployment, we calibrate the robot using the nyrio-one-web GUI, and let the visitors use a joystick to pick-and-place boxes with the robotic arm. Visitors will observe a smooth robot control when there are no control command losses.
(3) Then, we inject control command losses as the visitors drive the robot with the joystick, so visitors observe how the injected losses harness the robot stability.
(4) Finally, we enable FoReCo and continue the remote control of the robot, namely, the pick-and-place actions. Visitors will observe a smooth driving, and visualize in the InfluxDB GUI the lost commands, and the commands recovered by FoReCo.

During all the demonstration we use the GUIs in Figure 2 to deploy the NS (BASS GUI), to calibrate and remotely drive the robot (nyrio-one-web GUI), and to visualize the control commands recovered by FoReCo (InfluxDB GUI). It is worth remarking that visitors will use the joystick in the demonstration to pick-and-place boxes with/without command losses in steps (2) and (3), respectively. Therefore, we propose an interactive demonstration to easily understand the problematic solved by FoReCo, i.e., the loss of control commands that guide a robotic manipulator.

ACKNOWLEDGEMENTS

This work has been partly funded by the European Commission through the H2020 project Hexa-X (Grant Agreement no. 101015956), and the Spanish Ministry of Economic Affairs and Digital Transformation and the European Union-NextGenerationEU through the UNICO 5G I+D 6G-EDGEDT and 6G-DATADRIVEN.

REFERENCES

[1] 5G ACIAS. 2019. *5G for Connected Industries and Automation.* White Paper.
[2] Doris Aschenbrenner, Michael Fritscher, Felix Sittner, Markus Krauß, and Klaus Schilling. 2015. Teleoperation of an Industrial Robot in an Active Production Line. *IFAC-PapersOnLine* 48, 10 (2015), 159–164. https://doi.org/10.1016/j.ifacol.2015.08.125 2nd IFAC Conference on Embedded Systems, Computer Intelligence and Telematics CESCIT 2015.
[3] Michel Franken, Bert Willaert, Sarthak Misra, and Stefano Stramigioli. 2011. Bilateral telemanipulation: Improving the complementarity of the frequency-

and time-domain passivity approaches. In *2011 IEEE International Conference on Robotics and Automation*. 2104–2110. https://doi.org/10.1109/ICRA.2011.5979969

[4] Milan Groshev, Jorge Martín-Pérez, Carlos Guimarães, Antonio de la Oliva, and Carlos J. Bernardos. 2022. FoReCo: a forecast-based recovery mechanism for real-time remote control of robotic manipulators. *IEEE Transactions on Network and Service Management* (2022), 1–1. https://doi.org/10.1109/TNSM.2022.3173436

[5] Carlos Guimarães, Milan Groshev, Luca Cominardi, Aitor Zabala, Luis M. Contreras, Samer T. Talat, Chao Zhang, Saptarshi Hazra, Alain Mourad, and Antonio de la Oliva. 2021. DEEP: A Vertical-Oriented Intelligent and Automated Platform for the Edge and Fog. *IEEE Communications Magazine* 59, 6 (2021), 66–72. https://doi.org/10.1109/MCOM.001.2000986

[6] Christian Ott, Jordi Artigas, and Carsten Preusche. 2011. Subspace-oriented energy distribution for the Time Domain Passivity Approach. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 665–671. https://doi.org/10.1109/IROS.2011.6094697

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[8] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, Andrew Y Ng, et al. 2009. ROS: an open-source Robot Operating System. In *ICRA workshop on open source software*, Vol. 3. Kobe, Japan, 5.

[9] Da Sun, Fazel Naghdy, and Haiping Du. 2014. Transparent four-channel bilateral control architecture using modified wave variable controllers under time delays. *Robotica* -1 (07 2014), 1–17. https://doi.org/10.1017/S026357471400191X

[10] Da Sun, Fazel Naghdy, and Haiping Du. 2016. Wave-Variable-Based Passivity Control of Four-Channel Nonlinear Bilateral Teleoperation System Under Time Delays. *IEEE/ASME Transactions on Mechatronics* 21, 1 (2016), 238–253. https://doi.org/10.1109/TMECH.2015.2442586

[11] Da Sun, Fazel Naghdy, and Haiping Du. 2017. Neural Network-Based Passivity Control of Teleoperation System Under Time-Varying Delays. *IEEE Transactions on Cybernetics* 47, 7 (2017), 1666–1680. https://doi.org/10.1109/TCYB.2016.2554630