Adaptive Motion Control Middleware for Teleoperation Based on Pose Tracking and Trajectory Planning

Andreas Blank , Engin Karlidag, Lukas Zikeli, Maximilian Metzner and Jörg Franke

Abstract

Concurrent with autonomous robots, teleoperation gains importance in industrial applications. This includes human–robot cooperation during complex or harmful operations and remote intervention. A key role in teleoperation is the ability to translate operator inputs to robot movements. Therefore, providing different motion control types is a decisive aspect due to the variety of tasks to be expected. For a wide range of use-cases, a high degree of interoperability to a variety of robot systems is required. In addition, the control input should support up-to-date Human Machine Interfaces. To address the existing challenges, we present a middleware for teleoperation of industrial robots, which is adaptive regarding motion control types. Thereby the middleware relies on an open-source, robot meta-operating system and a standardized communication. Evaluation is performed within defined tasks utilizing different articulated robots, whereby performance and determinacy are quantified. An implementation sample of the method is available on: https://github.com/FAU-FAPS/adaptive_motion_control.

Keywords

Industrial Robots · Teleoperation · Motion Control

A. Blank (✉) · E. Karlidag · L. Zikeli · M. Metzner · J. Franke
Institute for Factory Automation and Production Systems (FAPS), Friedrich-Alexander-Universität
Erlangen-Nürnberg (FAU), Erlangen, Germany
e-mail: andreas.blank@faps.fau.de

J. Franke
e-mail: joerg.franke@faps.fau.de

1 Motivation

While conventional robot applications mainly automate repetitive tasks in known environments, the relevance of robotics in applications with existing uncertainties is gaining importance in a variety of industries. This includes among others bin-picking tasks in production or intralogistics, nuclear waste handling in decommissioning, mining or oceanographic operations, assistance in healthcare as well as aeronautics and space missions [1–5]. Autonomous robot capabilities are potential enablers to accomplish these tasks [6], but due to uncertainties, still failures and system downtimes occur frequently [7]. In contrast, teleoperation already provides a more reliable robot application in complex scenarios with existing uncertainties [8, 9] and also enables efficient remote intervention for autonomous systems [7]. In addition, teleoperation increases safety at work in hazardous areas, as operators can act at a distance [5, 10]. Due to technological advances in most robotic fields, the demand for teleoperation applications have increased significantly in the last decade [4].

The appropriate translation of operator inputs into robot movements has a key role in teleoperation. Common teleoperation systems for industrial robots either control the robot in cartesian space, through joint angles or by six degrees of freedom pose goals with subsequent trajectory planning. Often these approaches share a lack of reactivity [11–13]. In contrast, approaches optimizing this characteristic through a reactive pose tracking motion control are often proprietary, vendor-specific solutions, resulting in a restricted interoperability [14]. However, for teleoperation in individual remote handling tasks with varying characteristics, interoperability as well as providing different motion control modes is beneficial. This includes reactive pose tracking, jogging control as well as robot movement based on collision-free trajectory planning. While pose tracking allows precise operator control in confined spaces, the latter enables task-specific programming, supervised automation and thus the relief of operator workload.

Our paper focuses motion control for teleoperation of industrial robots especially in scenarios with limited pre-defined circumstances, such as intervention of failed autonomous robot systems. We present a situation-dependent adaptive motion control middleware, featuring reactive pose tracking and collision-free trajectory planning for teleoperation. Thereby interoperability, a standardized communication as well as a simple extension by open-source robot libraries is addressed. The middleware is designed to support different Human Machine Interfaces (HMI) as input, while more intuitive and immersive experiences are achieved in combination with Virtual Reality (VR).

2 Related Work: Motion Control in Teleoperation Systems

Early teleoperation systems linked the controlling master and the controlled slave robot system mechanically [15]. Later, systems became computer-aided and software-intensive [16]. The increasing performance of robot controllers therefore provided a powerful base. Consequently, teleoperation tasks were performed through network-based information and command transmissions [17].

An interface utilized for remote control in industrial applications is the so-called jog-interface, enabling positioning of the manipulator based on incremental applied cartesian vectors or joint angles [18]. Joysticks or teach pendants are common as input devices. Disadvantages are the requirement of specific interfaces respectively an access to the robot controllers processing, limitations regarding an intuitive control operation and risk of missing the targeted pose. The latter results due to difficulties for the operator to recognize reaching the target and due to the time required by the robot to stop.

Remote control approaches of the last decades can often be characterized as proprietary solutions. Reasons therefore are mainly vendor-dependent differing motion control modes, communication interfaces and programming command sets. A distinction can be made between high-level and low-level approaches. While high-level approaches are based on standard robot controller instructions and communication interfaces [12], low-level ones utilize even more vendor-specific procedures and require more advanced programming [14]. These low-level approaches by contrast, enable a reactive and dynamic robot pose control, a requirement for appropriate teleoperation. Alternative methods for teleoperation are based on the well-established Robot Operating System (ROS), which offers middleware-typical functionalities as well as a simple extension of functionalities through open-source algorithms [13, 19, 20]. In particular, the MoveIt path planning extension including the Open Motion Planning Library allows defined target pose reaching. However, limitations derive from a lack of a dynamic and reactive pose tracking due to the time necessary for trajectory planning and the inability to redirect the robot during automated trajectory execution.

The above considered approaches differ on a lower layer regarding communication techniques. Network-based communication via Ethernet and the Internet Protocol is thereby common. At Transport Layer the User Datagram Protocol and the Transmission Control Protocol are utilized. Differentiation between approaches is possible in terms of data encoding standardization. Mentioned should be the Interoperable Teleoperation Protocol (ITP) [21] and the Simple Message Protocol (SMP) of ROS-industrial Edwards and Lewis [22]. While ITP is especially suitable in a standalone setup for relative motions in cartesian space, SMP supports a more complete range of motion types, is ROS-compliant and provides time-efficient communication with robot controllers.

In addition to the robot slave system, the HMI as input system also influences the motion control experience. For a suitable middleware it is important to support a wide range of HMIs. These include input devices as master arms, keyboards, teach pendants and gamepads in combination with single- or multi-monitor setups as well as VR-systems. Especially the latter ones show potential for increased operating intuitiveness and immersivity [23]. Contrary to approaches based on displacement of a master device within a small input area [19], these also enable remote guidance of the Tool Center Point (TCP) based on absolute poses, resulting in an more accurate target pose reaching. Furthermore, VR allows the augmentation by motion relevant information.

For teleoperating industrial robots in not fully defined environments, a middleware with certain characteristics, uniting and extending existing approaches is required. First, interoperability needs to be ensured through hardware abstraction and standardized communication. Due to the variety of expected tasks, different motion control types during teleoperation should be provided for an improved situation-dependent operation. Available should be a reactive, dynamic pose tracking to fulfill more complex movements, collision-free trajectory planning for relieving workload from the operator and as base for automated operations as well as a cartesian jogging for precise linear motions for performing assembly operations. Finally, the middleware should be complementable with modern HMIs for intuitive operation.

3 Adaptive Motion Control Middleware for Teleoperation

In the following sections our method of an adaptive motion control middleware for teleoperation of industrial robots is described. First, an overview on architectural level is given, followed by a description of the implemented motion control modes. Finally, the systematics for a situation-dependent adaptive switching of modes is explained.

3.1 Overview: HMI, Middleware and ROS-Adaptive Control Driver

The teleoperation system consists of three main components: HMI, Teleoperation Middleware and ROS-Adaptive Control Driver (ROS-ACD). An overview of the middleware architecture and its mechanisms is given by the following Fig. 1.

An HMI consists of an operator input and a feedback channel. Thereby the input device is used for target pose control and the feedback channel allows multimodal integration of the operator. For both channels, our middleware supports a variety of devices. In our architecture the HMI is linked to the slave robot system via the Teleoperation Middleware which is integrated to a computer system running ROS. The latter is chosen since it provides a comprehensive set of open-source software libraries and tools for robotic applications. Thereby, the ROS-Industrial extensions provide hardware compatibility through drivers for a wide range of industrial robot systems.

Fig. 1 Teleoperation middleware integration as technical architecture modeling

An essential element of our approach is the ROS-ACD, which provides different motion control modes for a specific robot system. The driver itself is an extension of existing SMP-based ROS drivers. It includes a motion server for performing motion instructions and a state server for feedback. A key feature is enabling reactive velocity control in contrast to common provided motions based on planned trajectories.

Our proposed middleware pursues to retain the existing ROS architecture for ensuring interoperability. It serves as an additional abstraction layer between ROS and the ROS-ACD, enabling functional extension, especially for providing different motion control modes: jogging, collision-free trajectory planning and pose tracking. Different interfaces are used for communication, e. g. the MoveIt-Interface serves to interchange target pose goals and planned trajectories. For dynamic pose tracking a specific controller is implemented. The Robot State Interface records the actual robot position for monitoring and control purposes, whereby all messages are also tunneled to ROS. The middleware is designed for usage in combination with the ROS Industrial Robot Client. Instead of a direct linkage to the robot driver, the client interconnects with the middleware. Thereby, data traffic is bi-directionally passed through the middleware.

3.2 Motion Control

The following section describes the motion control modes. Since jogging and trajectory planning are integrations of common methods, pose tracking will be described in detail.

Collision-Free Trajectory Planning. Collision-free trajectory planning is suited for safely approaching target positions, especially in environments with potential collision. This is of importance for teleoperation since it enables to reduce operator workload. It also serves the teleoperation system to provide autonomous robot capabilities, such as

supervised grasping of objects. As described, our approach is based on MoveIt for planning trajectories. The target pose thereby is specified via the HMI (e. g. VR-controller). As feedback from MoveIt the planned trajectory with intermediate robot settings is spatially visualized for the operator through the HMI. If released, the middleware performs the motion control for carrying out the planned trajectory.

Pose Tracking Control. Pose tracking is based on cartesian velocity control implemented within the ROS-ACD. It allows reactive remote guidance of the TCP based on a dynamic target pose. First an initial target pose is defined near to the actual robot TCP through the HMI. The subsequent operator movements are tracked, so highly individual splines can be performed. The actual implementation depends on the functions available through the vendor-specific Application Programming Interface (API). For reactive motion control, a deterministic real-time capable low-level function is required, which is preferably executed in each interpolation cycle T_{IP} of the robot controller.

If the API provides a native function for cartesian velocity control, this is preferably used to take advantage of existing safety features, such as limiting acceleration and velocity or singularity handling. Otherwise, position based incremental motion functions are used for the implementation. This is achieved by conversion of the velocity commands into incremental displacement for an interpolation cycle. A velocity command $\mathbf{v}_{cmd}$ is a 6D vector consisting of a linear and an angular component. Multiplying the vector by T_{IP} results in a relative, incremental motion vector $\Delta\mathbf{x}_{inc}$.

$$\mathbf{v}_{cmd} = \begin{bmatrix} v_x & v_y & v_z & \omega_x & \omega_y & \omega_z \end{bmatrix} \quad (1)$$

$$\Delta\mathbf{x}_{inc} = \mathbf{v}_{cmd} \cdot T_{IP} \quad (2)$$

For a closed loop, the pose controller requires the actual pose of the TCP as feedback. It is necessary to adapt the driver to transmit feedback messages synchronized to the interpolation clock which are used to trigger the pose control loop. This ensures the deterministic processing as well as the necessary frequency matching of the velocity commands. Since the driver only sends joint positions $\mathbf{q}$ as feedback, the actual pose $\mathbf{p}_{fbk}$ is calculated through the forward kinematics. The target pose $\mathbf{p}_{cmd}$ is provided by the HMI. A PID controller compensates the tracking error $\mathbf{e}$. The output can be limited to avoid unwanted high accelerations and velocities. Finally, the resulting velocity commands $\mathbf{v}_{cmd}$ are transmitted to the robot controller. The block diagram of the control loop is shown in Fig. 2.

Jogging Control. In addition to the two described methods above, the middleware provides a velocity-command based jogging of the robot. This serves to perform simple movements. For realization, one or more input-device joints of the HMI are accordingly mapped to robot joints. Motion control is either based on the individual joints of the robot via joint angles or in cartesian reference to the TCP.

Fig. 2 Block diagram of the pose tracking controller

3.3 Adaptive Control-Mode Switching

Depending on the task, it is beneficial to switch between motion control types while teleoperation. For this purpose, a state machine is implemented within the ROS-ACD. The switching is triggered by the reception of corresponding message types defined for the specific motion command. Provided modes can be used immediately without having to switch manually. This enables an adaptive, situation-dependent switching. For example, in combination with a VR-HMI, the distance between the actual TCP pose and the target pose serves as a threshold for the system to switch automatically. Furthermore, the middleware ensures a fail-safe switching through idle state monitoring.

4 Experimental Setup and Evaluation

In this chapter, our demonstrator system setup and experiments are described. A qualitative evaluation is performed based on a real-world teleoperation handling task. The results retrieved are backed up by a quantitative evaluation.

4.1 System Setup and Qualitative Evaluation

The system is realized according to Fig. 1. As HMI an existing VR-interface is used (cf. [8, 22]). The Teleoperation Middleware is integrated into ROS Melodic running on Ubuntu 18.04 LTS. The robots utilized are a Yaskawa HC10 with YRC1000 controller and a Stäubli TX60L with CS8C controller. While the Stäubli controller has a native function for velocity control, the ROS-ACD for the Yaskawa YRC1000 is extended according to our method. Within this implementation only the proportional part of the PID controller is used. Both robot controllers run with an interpolation cycle T_{IP} of 4 ms. For safety reasons during evaluation, velocity commands are limited to a maximum velocity v_{max} of 250 mm/s.

For qualitative evaluation, different operators perform handling tasks (see Fig. 3). Thereby, industrial components are picked out of small load carriers (SLC) and

Fig. 3 Teleoperated handling of components: Yaskawa HC10 (a) and Stäubli TX60L (b)

subsequently are positioned within an assembly fixture. As depicted, the HC10 is equipped with a two-finger parallel gripper and the TX60L with a magnetic gripper.

Due to the intuitive operation, the sensitive and accurate positioning, even in confined spaces (e. g. within the SLC or fixture), pose tracking is the most preferred motion control mode. After a few minutes of handling, the operators begin to use the collision-free trajectory planning mode as well to reduce time and required physical body movements. This mode is thereby used to bridge greater distances and for automated reaching of defined pose goals. The jogging mode in cartesian space is seen as suitable for linear joining operations. One reason for this is the possibility to simply lock other movement directions. The velocity control of the CS8C is perceived as smoother compared to our implementation for the YRC1000. In terms of reactivity the latter performs slightly better. Both results from the more intensified filtering by the proprietary function.

4.2 Quantitative Evaluation of Pose Tracking Motion Control

In quantitative terms, pose tracking motion control is evaluated regarding reactiveness, processing determinacy, dynamics and accuracy. Analyzed are the velocity control and the higher-level pose tracking controller of the middleware. Evaluations are performed on the Yaskawa YRC1000/HC10 to verify the feasibility of our own velocity control implementation. The analyses are performed by recording target and actual states for position and velocity of a single axis (x-axis), whereby the behavior is valid for all axes.

The velocity control step response and trigger response time are shown in Fig. 4. The commanded velocity $v_{x,cmd}$ of 100 mm/s is reached after about 200 ms, while the settling time (with 2% error) is already reached at 175 ms. The observed feedback velocity $v_{x,fbk}$ results from the processing of the velocity commands on the robot controller as incremental motion. It is obvious that the robot controller limits the acceleration, which is reflected in the gradient of the graph. In addition, a small delay at the beginning is noted, which is due to communication, execution time and the internal dynamics of the robot system. However, the reactivity of the velocity control can be evaluated as sufficient for the addressed use cases.

Fig. 4 Velocity control step response (left) and trigger response time (right)

The evaluation of the right plot allows conclusions about reliable and deterministic command execution. The response time represents the duration between feedback arrival from the robot controller as trigger signal and the consequent transmission of the velocity command. On the robot controller side, feedback streaming as well as the execution of the last received velocity command is synchronized with the interpolation clock with a cycle time T_{IP} of 4 ms. Thus, the velocity command must be sent as soon as possible to be available on-time at the robot controller until the next cycle. Since the execution of velocity commands is delayed by one interpolation cycle, T_{IP} at the same time represents a system dead-time. The results show a maximum response time of 64.7 μ s with an average value of 29.7 μ s. The on-time provision of commands is also reflected by the velocity profile. As a result, it can be concluded that the velocity control ensures a robust and reactive motion control.

The pose tracking controller is evaluated with respect to dynamics. The step response and the dynamic behavior for different controller settings are investigated (see Fig. 5). It is obvious that higher controller gains K_P results in a faster controller behavior. However, values chosen too high ($K_P \geq 4$) tend to overshoot, which is not desirable in teleoperation. The linear gradient of the step response graphs at the beginning is characteristic, as it results from the limitation of v_{max} . While $x_{fbk,1}$ cannot reach the steady state after 5 s, the others stabilize within the range of 2–3.5 s. The noticeable minimal stationary error is due to the resolution of the increments, which is limited to 1 μ m. Since such a small error would result in an increment smaller than the resolution, it cannot be applied. However, depending on K_P the observed errors are within a range of 18–96 μ m. Consequently, these can be neglected for our application.

The right graph shows a trajectory performed via pose tracking. Here x_{cmd} corresponds to the target position and $x_{fbk,n}$ corresponds to the individual feedback position. Here v_{max} represents a limiting factor. Thus, the tracking error can be minimized by a balanced parameterization of the two values. Performed experiments show, that a teleoperation with a too small resulting tracking error is perceived as hypersensitive. For example, the resulting control behavior of $x_{fbk,1}$ was preferred by operators.

Fig. 5 Step response (left) and dynamic tracking (right) with different control settings

5 Summary, Conclusion and Future Work

In this work, a method of an adaptive motion control middleware for teleoperation of industrial robots has been described. The resulting middleware improves interoperability and offers a variety of motion control modes to be chosen adaptively depending on the situation. This has proved to be beneficial for tasks in environments with existing uncertainties. The integration within ROS enables among other advantages the support of a wide range of robot systems. The evaluation results prove reactive pose tracking capabilities suitable for dynamic teleoperation in complex tasks. The performed handling experiments show the potential for intuitive operation, especially in combination with a VR-based HMI. Thereby, bandwidth requirements, system latencies and the suitability for long-distance teleoperation are detailed in Kohn et al. [23]. Future work will further improve the control behavior by using the entire PID-controller and low-pass filtering of HMI-inputs. In addition, we will implement collision avoidance in pose tracking mode. These and further research activities are addressed within the close to application research project VIRERO on volume-optimized conditioning of radioactive waste [5].

Acknowledgements Parts of this research were conducted within the project FORo-botics (AZ-1225-16), funded by the Bavarian Research Foundation (BFS). The authors would like to thank the foundation and the expert reviewers for their valuable advice.

References

1. Althoefer, K., Konstantinova, J., Zhang, K.: Position and velocity control for telemanipulation with interoperability protocol. In: TAROS 2019. Lecture Notes in AI. Springer, Berlin (2019)
2. Basso, A., Hlaváč, V., Hulka, J., Jilich, M., Krsek, P., Malassiotis, S., Molfino, R., Smutný, V., Wagner, L., Zoppi, M.: Towards intelligent autonomous sorting of unclassified nuclear wastes. *Procedia Manuf.* **11**, 389–396 (2017). <https://doi.org/10.1016/j.promfg.2017.07.122>

3. Blank, A., Hiller, M., Zhang, S., Leser, A., Metzner, M., Lieret, M., Thielecke, J., Franke, J.: 6DoF pose-estimation pipeline for texture-less industrial components. In: Bin Picking Applications IEEE European Conference on Mobile Robots (ECMR), Prague, Czech, pp. 1–7 (2019)
4. Blank, A., Havenith, A., Kohn, S., Querfurth, F., Zwingel, M., Metzner, M., Franke, J.: Robotic technologies for volume-optimized conditioning of radioactive waste—VIRERO. In: AiNT (Hrsg) 9th International Conference on Nuclear Decommissioning, Aachen (2020)
5. Blank, A., Berg, J., Zikeli, L., Lu, S., Sommer, O., Reinhart, G., Franke, J.: Intervention strategy for autonomous mobile robots. *wt Werkstattstechnik online* (Bd. 110 (Nr. 9)):601–606 (2020)
6. Buschhaus, A., Blank, A., Ziegler, C., Franke, J.: Highly efficient control system enabling robot accuracy improvement. *Procedia CIRP* **23**, 200–205 (2014). <https://doi.org/10.1016/j.procir.2014.03.200>
7. Buschhaus, A., Blank, A., Franke, J.: Vector based closed-loop control methodology for industrial robots. In: 2015 International Conference on Advanced Robotics (ICAR), pp. 452–458 (2015)
8. Dalvand, M.M., Nahavandi, S.: Improvements in teleoperation of industrial robots without low-level access. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC), San Diego, USA, pp. 2170–2175 (2014)
9. Edwards, S., Lewis, C.: ROS-industrial: applying ROS to industrial applications. In: IEEE International Conference on Robotics and Automation, ECHORD Workshop (2012)
10. Furuya, T., Yamashita, T., Kurisaki, T., Ikeda, H., Monji, T., Katoh, R.: Supervised robot system and computer aided teleoperation. In: IEEE International Workshop on Intelligent Robots, pp. 685–689. IEEE (1988)
11. Global Information Inc. Teleoperation and Telerobotics: Technologies and Solutions for Enterprise and Industrial Automation 2020–2025. Report. MindCommerce (2020)
12. Goertz, R.C.: Remote-control manipulator. US Patent No. 2,632,574 (1953)
13. Kawatsuma, S., Fukushima, M., Okada, T.: Emergency response by robots to Fuku-Shima-Daichi accident: summary and lessons learned. *Ind. Robot. Int. J.* **39**(5), 428–435 (2012). <https://doi.org/10.1108/01439911211249715>
14. King, H.H., Hannaford, B., Kwok, K.-W., Yang, G.-Z., Griffiths, P., Okamura, A., Farkhatdinov, I., Ryu, J.-H., Sankaranarayanan, G., Arikatla, V., Tadano, K., Kawashima, K., Peer, A., Schaub, T., Buss, M., Miller, L., Glozman, D., Rosen, J., Low, T.: Plugfest 2009: global interoperability in telerobotics and telemedicine. In: IEEE International Conference on Robotics and Automation, pp. 1733–1738. <https://doi.org/10.1109/robot.2010.5509422> (2010)
15. Kohn, S., Blank, A., Puljiz, D., Zenkel, L., Bieber, O., Hein, B., Franke, J.: Towards a real-time environment reconstruction for VR-based teleoperation through model segmentation. In: IEEE International Conference on Intelligent Robots and Systems, Madrid, Spain (2018)
16. Lima, A.T., Rocha, F.A.S., Torre, M.P., Azpurua, H., Freitas G.M.: Teleoperation of an ABB IRB 120 robotic manipulator and barrettHand BH8–282 using a geomagic touch X haptic device and ROS. In: Joint Conference on Robotics: SBR-LARS, Joao Pessoa, Brazil, pp. 188–193 (2018)
17. Obermeier, B., Treugut, L.: Bericht Lernende Systeme in lebensfeindlichen Umgebungen Plattform Lernende Systeme, München (2019)
18. Pires, J., Sá da Costa, M.G.: Object-oriented and distributed approach for programming robotic manufacturing cells. *Robot. CIM* **16**(1), 29–42 (2000)
19. Prexl, M., Zunhammer, N., Walter, U.: Motion prediction for teleoperating autonomous

- vehicles using a PID control model. In: 2019 IEEE Australian and New Zealand Control Conference, Auckland, New Zealand, pp. 133–138 (2019)
20. Trevelyan, J.P., Kang, S.-C., Hamel, W.R.: Robotics in Hazardous Applications Springer Handbook of Robotics, Bd 19, pp. 1101–1126. Springer, Berlin (2008)
 21. Wang, M., Liu, J.: A novel teleoperation paradigm for human-robot interaction. In: Turau, V., Weyer, C. (eds.) (Hrsg) 2004 IEEE Conference on Robotics, Automation and Mechatronics, pp. 13–18. IEEE, Piscataway, NJ (2004)
 22. Yamada, S., Nomura, T., Kanda, T.: Healthcare support by a humanoid robot. In: ACM/IEEE International Conference on Human-Robot Interaction, Daegu, Korea, pp. 1–2 (2019)
 23. Yuan, F., Zhang, L., Zhang, H., Li, D., Zhang, T.: Distributed teleoperation system for controlling heterogeneous robots based on ROS. In: 2019 IEEE International Conference on Advanced Robotics and its Social Impacts (ARSO), pp. 7–12. IEEE, Piscataway, NJ (2019)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.