Robotronics – Robot Mechatronics

Tamim Asfour, Samuel Rader, Pascal Weiner, Felix Hundhausen, Julia Starke, Cornelius Klas, Stefan Reither, Christian R. G. Dreher, Fabian Reister, Miha Dežman, Charlotte Marquardt

Abstract Building robotic systems requires a seamless integration of mechanics, electronics, embedded systems and software. The chapter provides insights into *the robotronics*, the robot mechatronics, for the design of integrated humanoid robot systems developed to perform tasks in real world environments. Selected use-case are used as examples for mechatronics design and control.

1 Introduction

Mechatronics unites many different fields of science and engineering in an interdisciplinary way [44], including electrical engineering, computer science, mechanical engineering and information technology. The term itself is a hybrid term, taken from mechanics, electronics and computer science and originated in Yaskawa, Japan in 1971 [11]. It extends mechanical systems with sensors and microcomputers to build components of intelligent systems. This includes design of actuators, embedded control, design and integration of sensor and communication [11]. Robot systems such as humanoid robots as shown in Fig. 1 feature complex mechatronics and require close cooperation of many engineering and scientific disciplines to design and build such systems. Among others, mechanical design specialists need to design body parts such as arms, hands, legs and heads and optimize the components to increase strength and decrease weight. In addition, electronic design specialists develop custom-designed and embedded electronic circuits for sensor data processing and control, which must be integrated in a very limited space. Further, specialized low-level software requires expertise in embedded programming to develop programs that run directly on custom electronic devices and connect different components through appropriate communication protocols. Finally, high-level control

The authors are with the Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Germany, e-mail: asfour@kit.edu

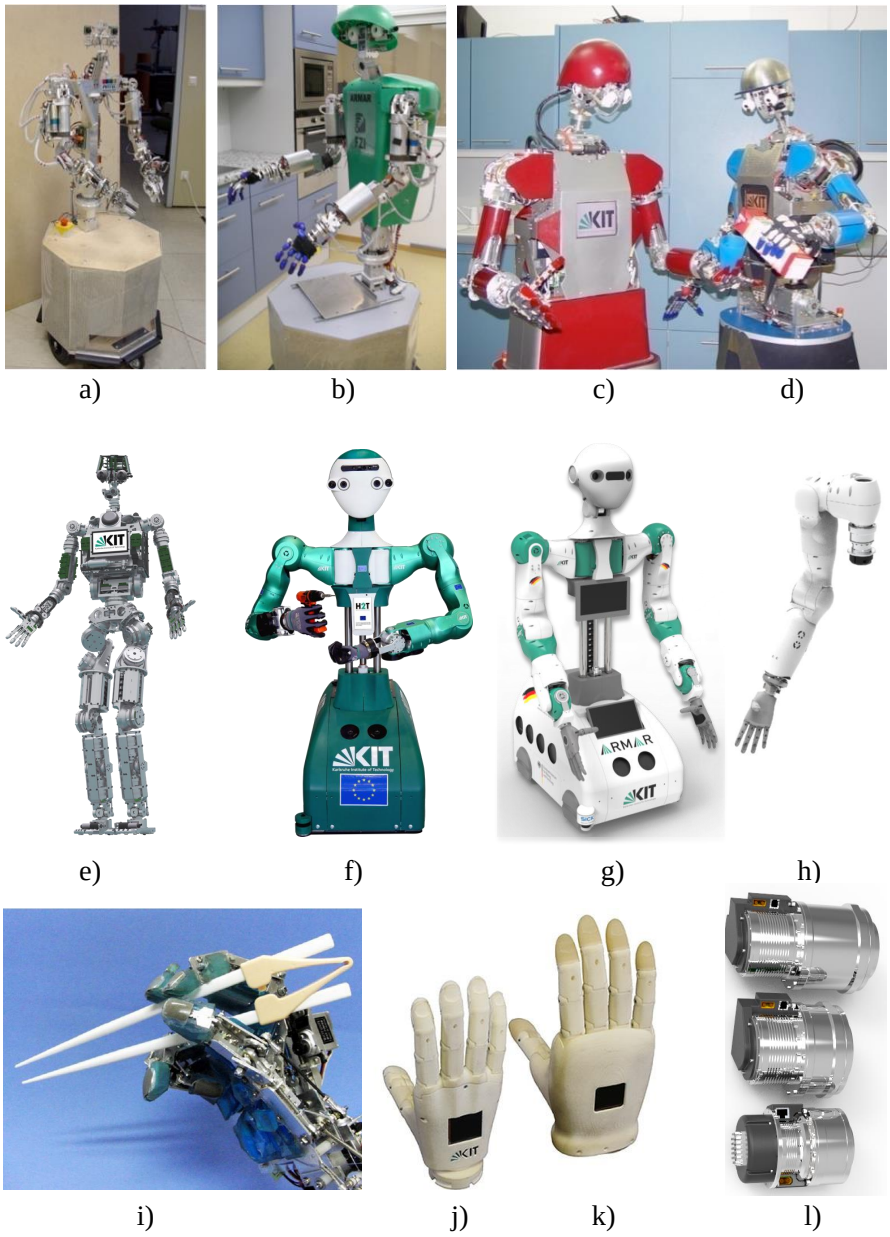

Fig. 1: Robotronics of humanoid robots and components. a) ARMAR-I, b) ARMAR-II, c) ARMAR-IIIa, d) ARMAR-IIIb, e) ARMAR-4 [4], f) ARMAR-6 [5], g) ARMAR-DE, h) ARMAR-7 arm, i) TUAT/Karlsruhe Hand [18], j) Female KIT Hand, k) Male KIT Hand, l) Sensor-actuator-controller units.

and communication design skills are required to robot software architectures that integrate symbolic planning and reasoning and sensorimotor control.

As such the mechatronics of robots, the robotronics, provide the embodiment for physical, embodied intelligence. This chapter starts with a short description of methodologies of mechatronic design and continues with several examples, taken from our research on humanoid robots at the Institute for Anthropomatics and Robotics in Karlsruhe Institute of Technology.

2 Design Approach

The large design space of mechatronic systems has led to emergence of many system design methods. A design method can help the engineers from different disciplines to facilitate smooth collaboration in the development of the mechantronic systems. Many such methods emerged at the end of the 20th century, like the waterfall model [7], the spiral model [9], and the V-model [17]. The V-model presents a general flow for the product development process, which starts with the identification of user's requirements and their decomposition into software and hardware. It continues with the fulfillment of functionality, the system integration of components and concludes with testing and user validation. The VDI guideline 2206 is a functional modeling methodology based on the V-model [22], which was developed and standardized by the VDI committee, the German engineers association. This standard provides guidance for managing the complexity of mechatronic systems.

It was developed in 2004 and builds on the good practices of previous software-based guides and represents the logical relationships of facts and interactions that form the foundation for the successful development and deployment of complex technical systems. The guide divides the design of mechatronic systems into four main phases, namely "System Design", "Domain Specific Design", "System Integration", and "Performance Check". The development process starts with the definition of requirements. Figure 2 shows our research areas at H^2T, which guide the design of our humanoid robot development.

The design of our humanoid robots is driven by the requirements related to the abilities of perception, grasping and manipulating objects as well as to skills from humans and experience. We take inspiration from the anatomy and motion behavior of the human body and deign integrated components, subsystems and complete humanoid robot systems. We consider continuous integration and testing crucial to expose design flaws and weaknesses, to improve the systems performance, and to make use of gained new knowledge for the design of new robot generations. Once the requirements are identified, the "System Design" defines a cross-discipline solution concept for the system. In this phase, the overall function of the system is divided into sub-functions. Several guiding tools or design principles are employed here to define clean interfaces between software and hardware components, to improve their robustness and ease the final system integration, see Fig. 3.

An important aspect in our design approach is *modularity*, where the mechatronic system is decomposed into modules to reduce the complexity of the robot and to make developed components reusable. Such modularity also allows easier fault identification, since the underlined modules may be isolated and tested outside of the system. A good example are the sensor-actuator-controller units (SAC units), which are used for the design of the arms of ARMAR-6. These units integrate the whole mechanical drive train with sensors and electronics for control and communication [42]. These SAC units can be *scaled* to three different power levels, but they retain a similar sub-assembly structure. Careful design and calculations are necessary to ensure the performance of the SAC unit function, especially the critical functions, like emergency shut down. The development is supported by modeling and analysis of system characteristics with the help of models and computer-aided tools for simulation. *The custom electronics circuits* connect power electronics, con-

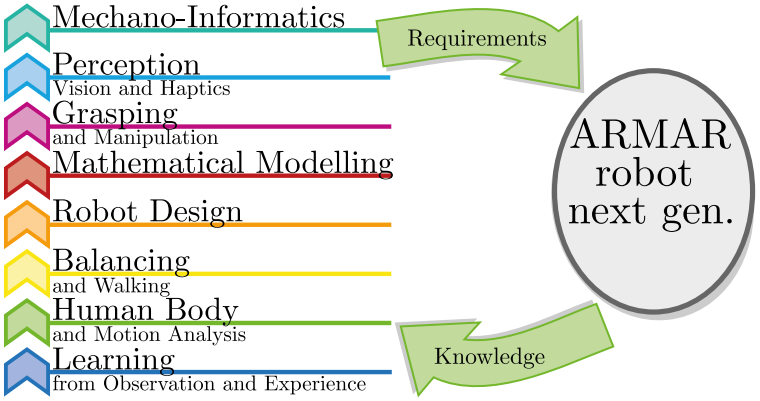

Fig. 2: Co-development of requirements, knowledge and real prototypes in the design of ARMAR robots.

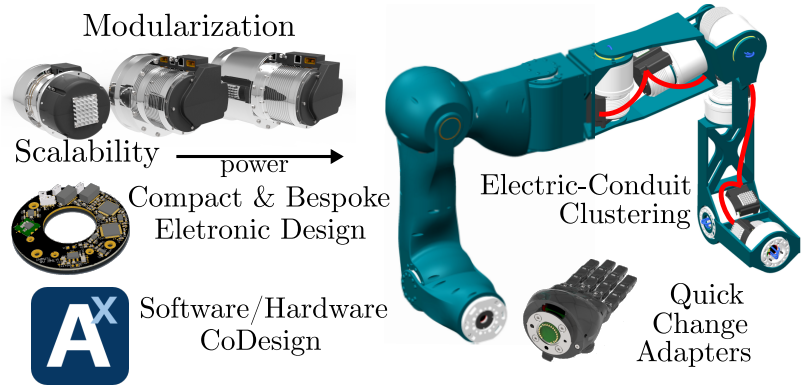

Fig. 3: Common design principles in development of ARMAR robots.

trol and communication lines into a compact PCB design to minimize the number of electronic conduits and increase their robustness. These PCB circuits have a custom shape that fits into the actuation unit resulting in a highly integrated system. The SAC units are placed throughout the robot and connected to the power and communication electrical conduits. The number of electrical conduits is minimized and *clustered* together so that only one power and one communication line connects the neighboring SAC units. The design of modular components also includes design of specialized interfaces, both on the software level and hardware level. A good example are the special *quick-change adapters* that allow quick replacement of robot hands to increase the robot's versatility. The adapter ensures a rigid connection between the hand and the arm, and connects the power and communication signal lines. The adapter also allows easy isolation of the component and testing.

The described domain specific phases are flanked by the "Modelling and analysis" of the system characteristics with the aid of models and computer-aided tools for simulation [58]. Computer Aided X (CAx) applications are used to support the product development process, generating data that is typically stored in a Mechanical Product Data Management (M-PDM) system, while electrical and electronic designers use Electrical/Electronic Engineering Solutions (EESs) to create data that is stored in Electrical PDM (E-PDM). Software designers use development solutions to create source code, which is managed through Software Configuration Management (SCM) or the Concurrent Versioning System (CVS). To reach the final product, each phase of product definition should be tested in system integration and concluded with system validation. In the specific case, the robot components are assembled together, and the joint design of software and hardware ensures that information about the status of components is readily available at all times. More information on system integration and validation is given through specific examples presented throughout the chapter.

3 Hardware Design

In order to describe the development of the hardware of mechatronic systems in detail, selected subsystems from our humanoid robots are presented in the following: Humanoid arms, heads and hands. After a general introduction to the design of these robot systems, the humanoid robots of the ARMAR family (Fig. 1) are used as examples to describe how such systems can be implemented.

3.1 Humanoid Arm Design

In contrast to industrial robot arms, humanoid robot arms are designed to be used for a wide variety of tasks, often in cooperation with humans. This results in higher demands on the robots kinematic structure as well as on the sensor setup, which must

also ensure the safety of humans. This section describes a possible development procedure using the ARMAR robots as an example: After defining the workspace, the appropriate actuators and sensors are selected. Then the mechanical arm structure is described. Finally, the development of a human-like wrist follows, which is particularly challenging.

3.1.1 Workspace Consideration

Joint Configuration: The most important function of a robot arm used to physically manipulate its environment is to bring an end-effector into a desired pose in the workspace. A pose describes position and orientation in R^3. To bring the end-effector into an arbitrary pose in the workspace, at least 6 degrees of freedom (DoF) are required, 3 of which are for the position and 3 for the orientation. Since industrial robots are typically used for the same task all the time, they can be designed with 6 or fewer number of DoF. However, in order to use humanoid robots for a wide variety of mobile manipulation tasks similar to humans, they are often designed with 7 or more DoF. Such redundant robot arms fulfill additional constraints given by the task such as collision avoidance or the generation of human-like movements.

ARMAR-III therefore has arms with 7 DoF each. Each arm consists of one spherical joint (3 DoF) in the shoulder and one in the wrist, and a rotational joint in between to allow flexion/extension of the elbow. This results in a large workspace that allows the execution of versatile manipulation tasks in a kitchen environments. To further increase the workspace, the arms of ARMAR-4 and ARMAR-6 each have an additional inner shoulder joint, resulting in 8 DoF arms. The advantage of these additional joints was investigated early in the design phase through simulations of reachability and manipulability [51]. Thus, with ARMAR-6, the bimanual workspace increases from 1.8 m^3 to 4.9 m^3 due to an additional inner shoulder joint on the left and right side.

When defining a joint configuration, kinematic singularities should also be avoided. It is possible to avoid singularities by adding displacements, for example between the upper arm axis and the extension/flexion joint of the elbow. In ARMAR-6, the elbow joint was moved forward by 55 mm for this purpose. In addition to avoiding singularities, this also increases the maximum angle for elbow flexion rotation.

Arm Proportions: Apart from the joint configuration and the joint limits, the lengths of the individual segments of the kinematic chain influence the working space. The first step should be to determine the required reach of the arm. It will ultimately determine how strong the first drives have to be and how heavy the entire arm will be. While the arms of ARMAR-III and ARMAR-4 have human proportions, the proportions of ARMAR-6 are much larger. To ensure that the robot is able to manipulate objects at a height of 2.3 m and at the same time pick up objects from the ground, an arm with a length of about 1000 mm from the center of the shoulder to the Tool Center Point (TCP) was developed. The Tool Center Point of the arms is located in the middle of the palm. With an inner shoulder joint, it even has

a reach of 1300 mm. In combination with a prismatic joint in the torso that can be moved by 400 mm, the robot is thus able to fulfill both conditions. The arm length of ARMAR-6 significantly exceeds the arm length of ARMAR-4, which measures 610 mm from shoulder to TCP and 810 mm including the inner shoulder joint. The segment lengths of the upper arm and forearm and the distance between the wrist and TCP for ARMAR-6 are based on data at the 95th percentile male. They are scaled up by a factor of 1.3.

3.1.2 Electromechanical Key Components

Actuation and Power Transmission: Apart from the kinematics and the resulting workspace, the maximum payload is one of the most important requirements for a robot arm. Based on this minimum load specification, the estimated masses and inertia of the arm segments, and the desired angular accelerations, the necessary torques in the individual joints can be calculated. Since motors with the desired joint torque are usually too large and heavy, they are usually translated by gear mechanisms. A wide variety of gears can be used in robot arms. Planetary gears and harmonic drive gears can usually be attached directly to the motor. The motor axis corresponds to the output axis of the gear unit. Other gear mechanisms such as spur gears, worm gears, belt drives, wire rope drives, linear drives and parallel kinematics allow the motor axis not to correspond to the joint axis. The motor can then be spatially separated from the arm joint. This principle has been used several times in the ARMAR-III arms.

The power transmission in the shoulder joints of ARMAR-III (Fig.4) is realized by a combination of belt drives and worm gears. The elbow is cable-driven and equipped with a complex cable roll mechanism for the transmission of mechanical energy to the following joint. The elbow motor is located in the torso of the robot. Both gear mechanisms allow the motors to be placed more proximal, i. e., closer to the body. This makes the arms significantly lighter. In addition, the arm proportions are not dependent on the size of the motors.

In contrast to ARMAR-III, the arms of ARMAR-4 and ARMAR-6 use slim brushless direct current motors (BLDC motors) and harmonic drive gears. This makes it possible to accommodate the entire drive train in the joint axis, which was used consistently in both robots up to the forearm. By concentrating most electromechatronic components in the joints, the structure between the joints as well as the cabling can be significantly simplified. This can ultimately lead to better robustness.

Sensors: Besides actuators, sensors are also required to control the robot arm. Position encoders are required as standard. In addition to an incremental or hall encoder on the motor shaft, modern robot arms also have an absolute encoder on the output of the gear unit. While incremental encoders only measure the relative rotation of the motor to its initial position in the form of ticks, the absolute encoder can determine the current joint position at any time.

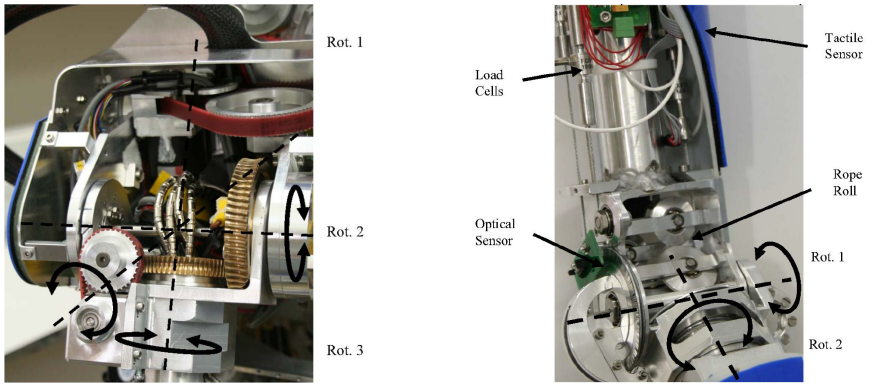

Fig. 4: Shoulder (left) and elbow (right) of ARMAR-III [1]

For robots used for physical human-robot interaction, torque control is critical to ensure humans safety. To achieve this, torque sensing is needed. Torque can be measured in a variety of ways. A very accurate sensor is obtained by applying shafts or spoke wheels with strain gauges. Alternatively, it is also possible to determine the torque using two high-precision absolute encoders, placed placed at the beginning and end of the output shaft. The difference between the two measured angles corresponds to the twist of the shaft, from which the torque can be calculated. Finally, the motor current can also be used to roughly determine the approximate torque. Since sensor data on forces and torques are particularly helpful when gripping objects, force-torque sensors in the wrists of robots are often used in addition to torque sensors in the joints. ARMAR-III, ARMAR-4 and ARMAR-6 all have a 6D force-torque sensor in the wrist, which allows forces and torques to be measured in all spatial directions and orientations. Other sensors frequently used in robotic arms are temperature sensors, which are used to monitor the heat in the arm and to initiate measures at critical threshold values to protect heat-sensitive components. Furthermore, temperature measurements can be used to compensate for temperature drift of other sensors. In robot arms, all conceivable other sensors can be accommodated. Another example are inertial measurement units (IMU), which combine several inertial sensors such as accelerometers and angular rate sensors. But also cameras, distance sensors and pressure sensors are an option.

3.1.3 Sensor-Actuator-Controller Units

As shown by the arm of ARMAR-III (Fig.4), actuators, sensors and motor controllers were often placed separately and scattered throughout the robot arm. While this has the advantage of utilizing every space in the arm, it makes design, assembly and maintenance extremely complex. The many distributed interfaces make it dif-

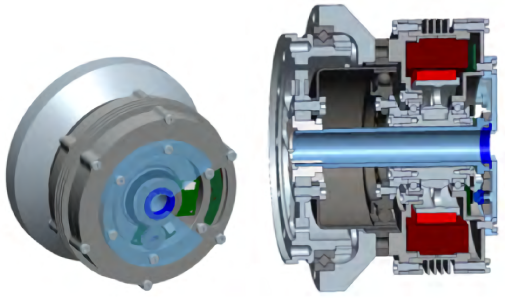

Fig. 5: Sensor-actuator unit of ARMAR-4 integrating motor, gearbox, incremental and absolute position sensors, torque and temperature sensors [4]

ficult to build a robust system, especially with regard to cabling. Furthermore, this rarely allows arm designs to be quickly reusable.

As a result, sensor-actuator units (SA unit) that integrate the drive train and sensors in one compact module with a well-defined interfaces have become increasingly popular in recent years. Other names for such modules are drive unit or joint unit. Fig. 5 shows a sensor-actuator unit as used in the ARMAR-4 arm and leg joints. They integrate a high performance BLDC motor, a high-ratio harmonic drive gear, an incremental encoder, an absolute encode, a torque sensor and temperature sensors. However, the motor controllers are placed outside the units, so there are several cables to be routed between the SA unit and the motor controller.

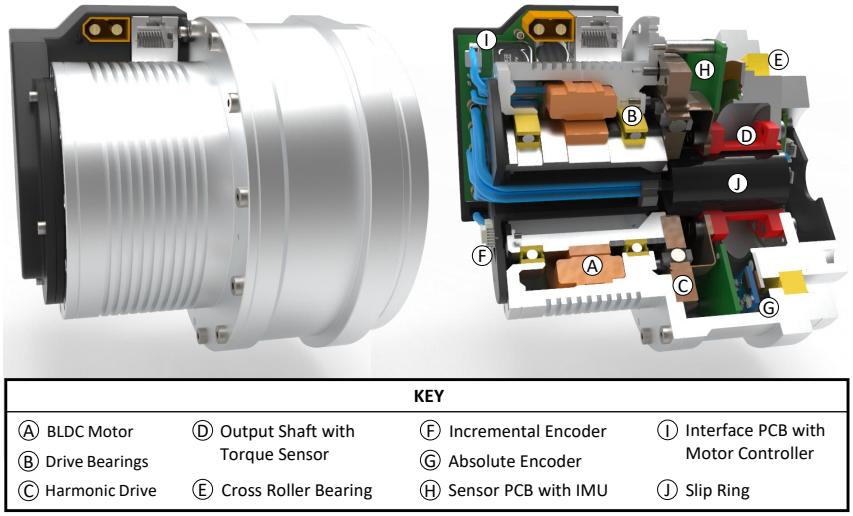

Fig. 6: Sensor-actuator-controller unit (SAC unit) of ARMAR-6

Therefore, the joint units for the arms of ARMAR-6 go even further. They integrate all important mechatronic components for actuation and control of individual joints into one module, the Sensor-actuator-controller units (SAC units), which is shown in Fig. 6. In addition to the drive train and the sensors of the ARMAR-4 joint units, they also contain a motor controller as well as electronics for control and communication via EtherCAT, a high-speed Ethernet-based fieldbus system that allows for periodically sampling of all sensors at 1kHz. Other additional components include an IMU on the sensor PCB of the SAC unit and a slip ring at its center. Slip rings are cable connections that can be rotated infinitely by sliding contacts consisting of gold rings and brushes. Slip rings allow all joints with SAC units to be rotated continuously without breaking the cables. Therefore, the upper or lower arm of ARMAR-6 can also be rotated infinitely and have no joint angle limits. In addition, slip rings make the cabling much more robust.

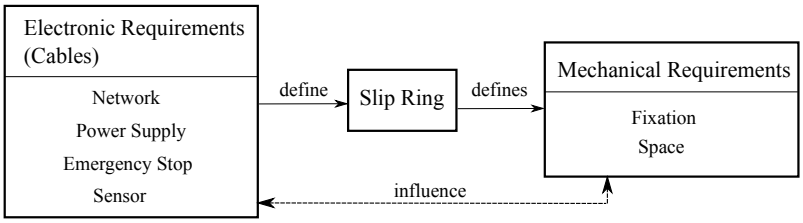

Fig. 7: Dependencies between different mechatronic parts of a SAC unit with the slip ring as an example [42].

The slip ring is a good example of the fact that in the development of highly integrated mechatronic components, electrical and mechanical design cannot be done separately. As shown in Fig. 7, electronic requirements define which slip rings are to be considered. How many cables are needed? How much current flows through the cables? Do shielded data cables have to be used? Based on these cabling requirements the size of the slip ring is different. Hence, the construction space and the fixation of the slip ring influence mechanical parts. The slip ring is of course only one component of the SAC unit. Since there are also many other dependencies between the mechanical and electronic components, it is rather a network of dependencies. In order to nevertheless create a robust, highly integrated mechatronic system, care was taken early to consider the electronics in the CAD design and vice versa. All cables with their bending radii were modeled in CAD and it was checked at an early stage whether there was sufficient space for the electronic components on the PCB.

In parallel to the CAD design, the electronic setup of the SAC units was developed (Fig. 8). In total, there are 6 circuit boards in the SAC unit, 5 of which were self-designed to improve the integration of the electronics through an optimal form factor. The connection between two SAC units electronically consists of only 3 cable connections: Supply voltage, a network line for EtherCAT and optionally

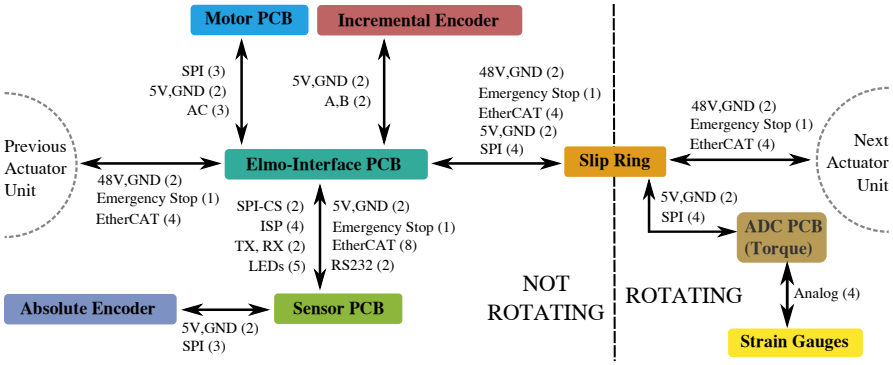

Fig. 8: Electronic setup of a SAC unit: PCB and cabling overview [42].

an emergency line. Multiple SAC units can be easily daisy chained together. This focus on a minimum of well-defined interfaces was also realized mechanically: In addition to two screw flanges, the units can also be mounted by a clam ring connection. To meet the different, installation space, torque and speed requirements of the individual arm joints, SAC units were developed in 3 sizes providing a maximum torque of 64 N m (wrist), 123 N m (elbow) and 176 N m (shoulder). With these units, each arm of ARMAR-6 has a payload of 11 kg at long range.

3.1.4 Arm Structure

The next step is to design the arm structure. There exist two basic concepts for arm structures: Frame construction and exoskeleton design (Fig. 9). In a frame construction, very simple flat parts or profiles are screwed together (Fig. 9, left). They are easy to manufacture and therefore low-cost. Many parts can be reused in the sense of a modular system and changes can be made quickly. Thus, additional sensors and other components can also be easily attached later. To cover and protect cables and electronics, additional cover parts can be attached to the frame structure.

In contrast, exoskeleton designs use complex shell parts that serve both as load bearing structure and as covering parts. This makes the parts difficult to manufacture and expensive, but in addition to saving on covering parts, this offers several advantages. Overall, there is more freedom in the design, which makes the structure close to the joint units less bulky and simplifies higher joint limits. Furthermore, the use of a hollow structure and the resulting moment of inertia is optimal for the different load cases, which a robot arm has to withstand. Hence, lightweight design is supported. And finally, such design makes assembly and maintenance easier and lead to increased robustness, in particular with respect to cabling. For the dual arm systems of ARMAR-6 and ARMAR-7 in Fig. 9, the exoskeleton design is realized consequently thanks to the modularity of the SAC units. However, using complex

power transmission mechanisms as they are used in ARMAR-III would not allow such implementation.

3.2 Humanoid Hand Design

The human hand is a complex and versatile system that allows us to intuitively grasp and manipulate a wide range of objects with different shapes and functionalities. From a kinematic point of view, the hand is a very complex system with 27 bones, 29 muscles and 21 DoF. Such complex kinematics endows our hands with versatility and dexterity in both powerful actions as well a fine-granular manipulation tasks. When building humanoid robotic hands however, the kinematic complexity poses great challenges for mechatronic integration and control. The actuation of the many degrees of freedom of humanoid hands requires high integration and smart actuator design within the tight space constraints of the human palm. In addition, the closing motion of the individual fingers and joints requires a well coordinated control to accomplish a stable grasp tailored to the object's shape and the desired manipulation goal.

Findings from neuroscience indicate that humans do not control every joint of the hand individually. Instead, correlations can be seen within the hand's DoF. These correlations can be partially explained by the mechanical structure of the forearm muscles, which are connected to several joints by the hand's tendons [35], and are partially caused by correlated muscle activation originated in the human brain [2]. Thereby, the posture of the human hand can be described by a significantly lower number of parameters - the hand synergies. Overall, more than 80 % of the informa-

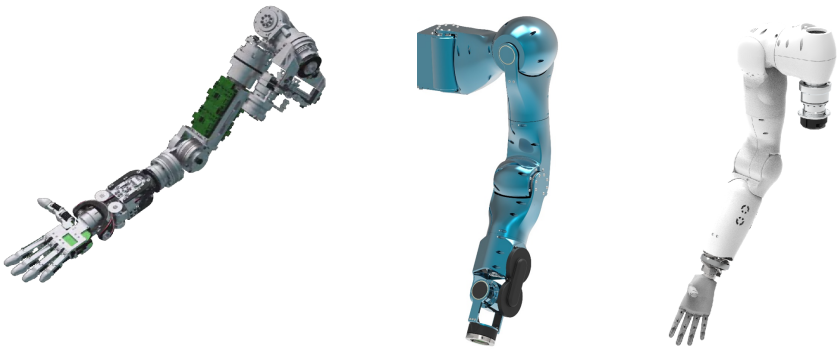

Fig. 9: Arm designs of humanoid robots: In contrast to the arm of ARMAR-4 [4] (left) that uses a frame construction, the arms of ARMAR-6 (middle) and ARMAR-7 (right) use an exoskeleton design.

tion transmitted by the joint angle configuration of a grasp posture can be described by two to three synergies [45].

An implementation of the hand synergies in hardware allows to accommodate both the high control complexity as well as the restricted integration space. The concept of underactuation reduces the number of *degrees of actuation* (DoA) by driving several DoF with a single motor. Thereby, an underactuation mechanism allows the implementation of hand synergies into the mechatronics of a humanoid robotic hand. In the following, a wide range of hand actuation strategies will be discussed including hydraulic actuators, underactuation based on electric motors as well as entirely soft kinematic structures. In addition, multimodal sensing and onboard computation for hand control will be focused.

3.2.1 Actuation Principles and Mechanisms

Most robotic hands are driven by electric motors or fluid actuators. In the following, both technologies are presented and it is explained how their power is converted and transmitted to actuate the various hand joints. Furthermore, softness in hands is discussed in more detail, through which grasping objects can be enhanced.

Fluid Hands: Originating in biomimetics, *flexible fluidic actuators* (FFAs) provide a suitable operating principle for compliant actuation. They are also often referred to as artificial (fluidic) muscles and provide a high power-to-weight ratio. The operating principle is based on the transmission of potential energy from the pressurized fluid within a flexible shell into mechanical force. While pneumatic actuators already come with inherent compliance due to the compressibility of gases, hydraulic actuators achieve compliance only with additionally integrated compliant structures [21]. In humanoid hands, the compliance of FFAs allows for adaptive grasping of several different objects [56].

In FFAs, torque can be directly generated from compression without the need for additional transmission elements. However, to be able to control the direction of the generate torque, as well as increase its magnitude and the range of motion, a guiding structure is required. This guiding element is essential for overall performance of the actuation [21]. In fluid driven hands the (mechanical) guiding structure generally comes with anthropomorphic features. Existing prototypes consist of a palm frame with attached artificial fingers. Single mechanical phalanges connected by artificial actuated soft joints form the fingers [56, 31].

Miniaturized hydraulic systems have been developed to be included into the palm of a humanoid hand without the need for external components except a power supply. Such a system generally consists of a hydraulic pump, a fluid reservoir, electric valves and an electronic unit which can be integrated into the palm of the hand. Choosing a suitable pump is mainly based on the size limitations. Gear pumps offer a feasible compromise regarding size, weight, efficiency and complexity. The FFAs, reinforced flexible bellows attached via solid fittings to the lever of the joint, are integrated directly into the joints. To ensure a stable grasp, a correct choice of hydraulic valves is significant. As the valves control the fluid supply of the FFAs,

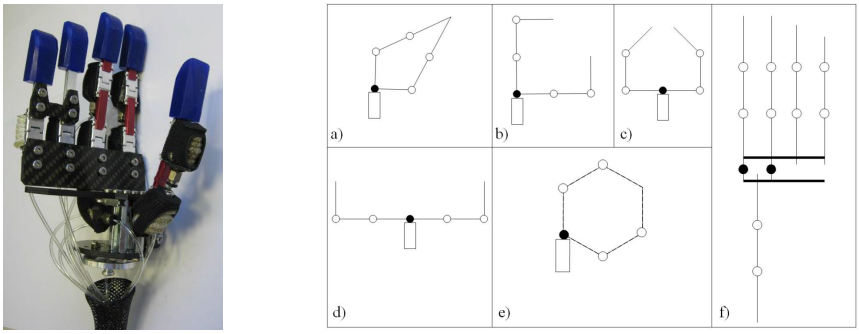

Fig. 10: The ARMAR-III pneumatic hand and its achievable grasp patterns [20]

only a complete closures and thus interruption of the fluid supply allow for specific positioning of the fingers during grasping. The modular structure of all components allows for independent actuation and positioning of the joints within the fingers [31].

Figure 11 shows an exemplary five-fingered pneumatic hand. It is a hybrid concept combining an anthropomorphic humanoid hand with a robotic gripper. While the rigid structure displays a more robotic appearance, the soft, inflatable joints in palm and fingers give it a humanoid look. To ensure precise grasping, a symmetric arrangement of all components was chosen. Mechanical joints are enhanced by reinforced flexible bellows such that a torque is applied by inflation. The compliant and lightweight system requires only an air supply and five wired cables for operation. Its modular design integrates all necessary parts into the hand, therefore no space is required proximal to the wrist and it can be attached to any robotic arm [20].

The compliant nature of pneumatic and hydraulic actuation, as well as the flexibility in actuator placement and pressure transmission facilitate the implementation of adaptive underactuation inspired by synergistic human grasping behavior. By the arrangement of fluidic chambers, actuation force can be distributed within several DoA. Communicating vessels connected to a single actuator can drive several fingers of a hand and automatically provide a balance in pressure and actuation force independent of the individual finger postures [48]. Further, pneumatically driven artificial muscles [47] allow for bio-inspired, human-like hand design. A robotic hand, that mimics human muscle actuation allows the direct transfer of human motion demonstrations into the robot control and provides insights on the biological functionality of the human hand [37].

Overall, fluidic hand actuation provides an alternative to electrically driven motors, that is more flexible in power transmission and actuator placement. The compressibility of the actuation fluid allows a humanoid hand to adapt to object shape and environmental constraints when complemented with compliant joint mechanisms or soft finger structures. By these means, grasp control can be simplified especially in highly restricted spaces. In such scenarios, the mechanical adaptation of fluid hands can directly cope with environmental constraints without the need to address them specifically in grasp control.

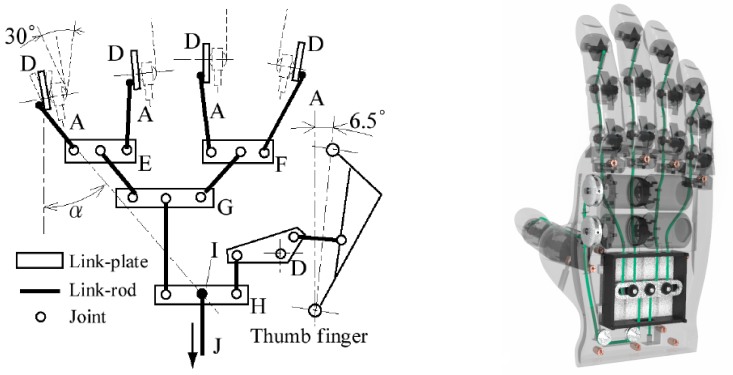

Fig. 11: Underactuation mechanism driving all five fingers of the hand with a single motor [19] (left) and the tendon routing of the male KIT Prosthetic Hand implementing the underactuation mechanism to drive the four fingers (adapted from [54]) (right)

Tendon-driven Hands: Inspired by the musculotendon system of the human hand, tendon-transmission of actuation forces is commonly used in humanoid robotic hands. The tendon's tractability allows for flexible force transmission from distantly placed actuation motors, usually positioned in the palm or the lower arm. However, tendons only transmit traction forces, thereby limiting the transmitted actuation to a single direction and two tendons or one stiff bar are required for the full actuation of one DoF [25, 26]. In the case of humanoid hands, high forces are required only in finger flexion to grasp objects, while finger extension merely needs to support the fingers own weight. Therefore, a widely-used strategy combines a tendon-driven actuation for finger flexion with springs locally integrated into the hand's joints to drive the finger's extension [16]. A hand with such a design is thereby actively closed, since the actuation of the finger flexion needs to counteract the extension force of the springs in the joints. Similar to the human model, the succeeding joints of one finger are commonly actuated together by a single tendon.

This coupled force transmission results in an underactuation of the robotic finger [34]. The flexible coupling by a tendon-transmission additionally allows the different finger joints within the kinematic chain to adopt different angles influenced by external forces applied to the finger phalanges. This adaptive underactuation causes a mechanical adaptation of the finger to the shape of a surface it closes onto. Similarly, adaptive underactuation can be applied to actuate two or more fingers by the same motor while still allowing their individual adaptation to an object's shape. This can be done either by routing a single tendon through several fingers [12] or with the use of an underactuation mechanism distributing the motor torque onto several finger tendons [19, 10, 8].

Figure 11 shows an underactuation mechanism based on a rigid bar transmission [19, 18]. An electrical motor is attached to rod J and the actuation force is distributed

equally to the thumb and the fingers respectively by the rocker H. The subsequent rockers G, E and F further divide the force equally between the rods A, that are actuating the finger flexion. All rockers, except for the transmission rocker D, are freely floating. While the hand is unobstructed, the entire mechanism is therefore pulled towards the motor, causing all fingers to close equally. If one finger is blocked by the object, the respective rocker tilts and the other finger continue closing.

A tendon-driven implementation of an adaptive underactuation mechanism is used into the KIT Prosthetic Hand [54]. The tendon routing throughout the mechanism as well as the entire hand is presented in Fig. 12. The hand is driven by two motors actuating the thumb and the fingers respectively. The thumb's tendon is directly attached to one motor and guided over two deflection pulleys in the thumb metacarpophalangeal and interphalangeal joints until it is attached within the fingertip. The second motor drives a tendon routed over the central pulley of the rocker within the underactuation mechanism and fixed at the housing thereafter. The rocker is sliding freely within the mechanism box and the motor tendon pulls the entire rocker bar down towards the palm, thereby distributing the motor torque to all four fingers. Two tendons connect index and middle finger as well as ring and little finger respectively. These tendons are routed over the deflection pulleys on either side of the mechanisms rocker. If one finger is blocked, the tendon rotates around the rocker pulley, allowing the other finger to continue closing. If both fingers connected by one tendon are blocked, the rocker tilts and thereby enables further motion of the other two fingers.

Along the fingers, the actuating tendon is routed over deflection pulleys in each joint and fixed at the fingertip, similar to the thumb. The flexion of the finger metacarpophalangeal and proximal interphalangeal joints is driven by this tendon, while the distal interphalangeal joint is fixed at an angle of 20°. Torsion springs in both joints provide a passive extension of the fingers. By varying the spring pretension for both joints, the correlation between the joint closing speed can be adjusted within the adaptive underactuation realized by the tendon. The difference in joint spring pretension thereby allows to mimic the human finger closing trajectory with a single DoA and a tendon transmission.

The underactuation mechanism has been adapted to actuate a different number of DoF as shown in [29]. By replacing the main rocker with a slider, a mechanism configuration to drive three fingers can be designed as shown in Fig. 12. Two fingers are still actuated by a single tendon, which is attached to both fingertips and routed over a pulley within the mechanism slider. This slider connects the finger tendon pulley with another pulley entangled by the actuating tendon from the motor. The third finger is directly actuated by the motor tendon. The combination of fixed and freely floating pulleys of this mechanism, which is similar to a gun tackle, ensures an equal force distribution between the uneven number of actuated fingers.

The KIT hands are applied both on humanoid robots and for prosthetics. They include a scalable finger design taking into account standardized components as well as electrical boards that can only be scaled stepwise. By these means, hands can be customized in different sizes corresponding to human hands for assistive robots or prostheses as well as larger than the human model for robots performing

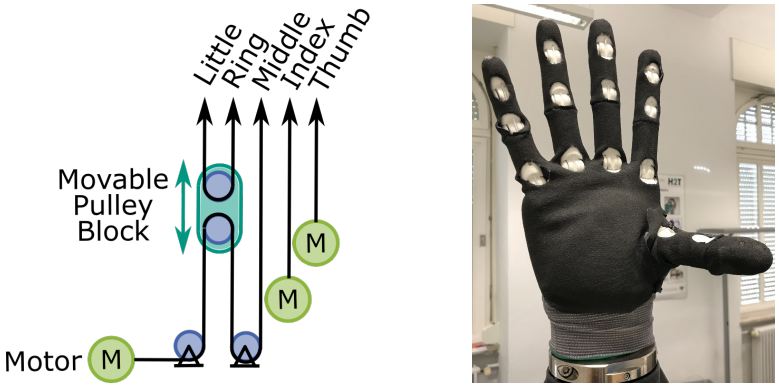

Fig. 12: Underactuation mechanism for three fingers actuated by one tendon (adapted from [29]) (left) and KIT Hand for the humanoid robot ARMAR-6 [3] (right)

taxing industrial work. The hands are able to grasp a wide range of objects found in daily household and workshop tasks, with more than 80 % grasp success rate. A cylindrical grasp force of more than 20 N allows the hands to lift and handle also heavy objects [54].

Soft Hands: Inclusion of soft elements into robots has become a very active research topic in the last two decades [41]. Soft surfaces, actuators and structures improve both safety and compliance of robots since collisions with the environment do not result in immediate damage. Especially for robotic hands this is desirable as these are specifically build to interact with the environment. Yet, the design of soft hands presents additional challenges since both state estimation as well as dexterous control are more difficult to archive for soft and deformable joints or fingers [13]. Since the structure of the fingers is soft, they do not only allow actuation in one direction, but bend and twist in multiple, potentially undesired directions. This can be mitigated by including rigid elements such as springs into the internal structure of the finger that limit deformation in undesired directions. Additionally sensors can be embedded into the fingers that are able to sense such deformation so that a suitable control mechanism can compensate for such movements.

The amount of softness introduced into robotic grippers varies greatly among different hands based on requirements and applications. In some applications soft materials are used on just a single joint to in order to increase compliance of contacts and increase mechanical robustness in case of collision [38]. In others the whole hand structure is exclusively constructed from soft material to exploit the compliance and adaptivity of the soft material to ease control [15].

In the following, we present an example, namely the design of the *KIT Sensorized Soft Hand* [53] and *KIT Finger-Vision Soft Hand* [29, 27], which illustrates the introduction of soft elements into the fingers of humanoid robotic hands. The hands are shown in Fig. 13.

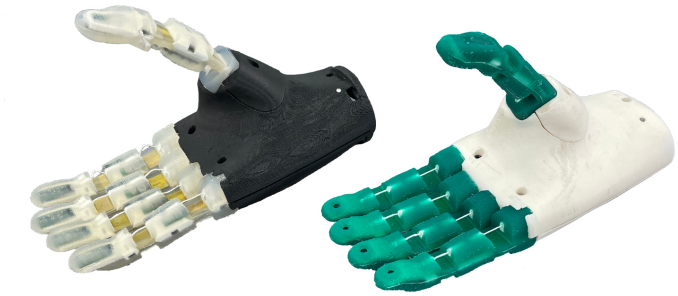

Fig. 13: The *KIT Sensorized Soft Hand* and *KIT Finger-Vision Soft Hand* [55]

Both hands share the same palm, including three motors. One motor drives the thumb, one the Index finger and the third motor drives the remaining fingers through an adaptive underactuation mechanism. The palm also houses an hybrid embedded system based on a combination of a microcontroller and a FPGA. Different fingers can be attached to the hand using a fixed interface, improving modularity of the design. The fingers for both hands share a similar mechanical design but differ in the included sensors. While the *KIT Finger-Vision Soft Hand* features a camera at the tip of each finger, the *KIT Sensorized Soft Hand* includes a multimodal haptic sensor system.

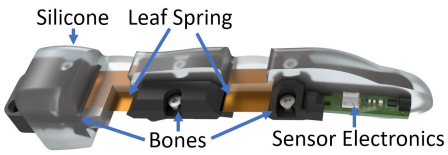

Fig. 14: The mechanical structure of the soft fingers. The silicone is cut after the proximal phalanx to expose the leaf spring and bones, the distal bone is also cut to expose electronics that can be embedded in the fingertip.

The fingers are constructed from 3D-printed bones for the proximal, intermediate and distal phalanx connected by a metal leaf spring, as can be seen in Fig. 14. The finger is actuated by a tendon in flexion direction by a tendon and passively by the leaf spring in extension direction. While the bones completely prohibit bending of the finger at the phalanges, the finger can bend at the areas between the bones. Due to the leaf spring, the fingers bend preferably in the flexion/extension direction while being substantially more stiff in all other directions. The leaf spring has a thickness of 90 µm and forms the neutral phase during bending of the joints. electrical cables, connecting sensors in the finger tip to electronics in the palm can be directly taped onto the leaf spring. This way the cables will mainly be subjected to a bending

motion with a defined bending radius and do not elongate during bending. *Flat flex cables (FFCs)* are especially well suited for this application. After the internal structure is assembled, the finger is inserted into a mold for casting. For casting, a two component silicone with a Shore A hardness of 13 is used.

3.2.2 Sensors and Embedded System

Very important for successful and reliable grasping and manipulation is the sensory information that can include tactile as well as visual information. The obtained information is required to provide feedback during the grasping process. Depending on the task, very high resolution tactile feedback in combination with visual feedback might be required, imagine the task of putting a thread through a sewing needle. In simpler tasks such as grasping of objects or manipulation, sensory information can be only necessary for detecting success and failure. While humans are not equipped with in-hand visual perception, in robotics visual feedback can be helpful for vision based control such as visual servoeing, where in hand vision can provide an un-occluded and better perspective onto the target object.

In the human body, sensory information is transmitted via the spinal cord towards the brain where decision are made, however typically in robotics data is acquired locally and due to the principal of modularity can also be processed locally in hand, allowing a well defined and simple communication interface to the hand.

Tactile Perception: Humans possess a sophisticated haptic perception system, enabling complex interaction with the environment. Likewise, haptic perception is a key enabling factor for dexterous grasping and manipulation in robotic hands. Yet, the robust integration of a rich sensor system into the confined space of robotic hands and fingers presents a unique mechatronic challenge [46]. The sensors need physical contact with the environment, making them especially prone to mechanical failure of the sensor structure or electrical connections. Additionally, the sensors need to be distributed throughout the mechanical structure of the hands since each part that is potentially in contact with the environment should ideally be innervated with sensors. This leads to a large number of required sensors and electrical signals that need to be routed through the mechanical structure of the hand.

Research on tactile sensors and their integration into robotic hands has gained a lot of traction in the last decade. Different sensor technologies have been explored to realize tactile sensors, including optical and visual, resistive, capacitive sensing, strain gauges, magnetic flux, vibrations, MEMS and the piezo-resistive/-electric effect [30, 39]. To illustrate the implementation of such a system into the fingers of a robotic hand, we will continue the example of the soft fingers of the *KIT Sensorized Soft Hand*.

The main idea behind the sensor system inside the soft fingers is to use readily available digital sensors and fabrication techniques to reduce fabrication effort and ensure reproducible results. Digital sensors offer the additional advantage that the analog signal of the sensing element is directly digitized inside the sensor and that the sensor usually offers a digital bus interface used for readout, minimizing wiring.

This follows the paradigm of encapsulation and modularity, allows for scalability of the design and offers the high robustness inherent to industrial sensing devices. The sensor system is multi-modal, including both dedicated normal force and separate shear force sensors, a proximity sensor, an accelerometer and custom joint encoders. All sensors are mounted on two PCBs, both connected to a central processing system in the palm through FFCs. An overview of the system is depicted in Fig. 15.

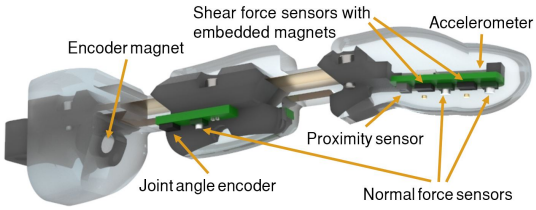

Fig. 15: Overview of the sensor system of the soft finger. The normal force sensor and joint angle encoder on the intermediate phalanx are duplicated the other side [53].

The shear force sensors are based on digital three-axis Hall effect sensors. To fabricate a shear force sensor, a magnet is placed in soft material above the sensor. When a force is exerted on the soft material, the magnet is displaced in relation to the sensor which causes a change in the measurement of magnetic flux measured by the sensor [33]. Co-planar movement of the magnet relative to the sensor represents a shear force while orthogonal movement represents the normal force component. While these sensors are able to measure shear forces well, they are not as sensitive to normal forces. Therefore, a second tactile sensing modality for normal forces is included in the sensing system.

The normal force sensors are based on digital miniature MEMS barometers. These sensors are cast into silicone which then acts as a transducer for the pressure applied to the surface of the finger onto the sensing element [50]. Depending on the hardness and thickness of the soft material, the sensitivity and sensing range can be adjusted. These sensors are able to detect weights as small as 0.5 g placed on the sensor and are hence well suited to detect initial contact between the fingers and the environment.

An infrared proximity sensor is located at the base of the fingertip to detect objects in close proximity. This allows sensing objects a few centimeters before they come in contact with the fingers and can for example be used in human-robot handover scenarios or to detect if the hand is close enough to an object to start grasping. Such proximity sensors can also be utilized as additional normal force sensors by measuring the deformation of the soft material above the sensor [57, 40]. Additionally, an accelerometer is placed on the back of the sensor PCB of the distal phalanx to detect sudden acceleration spikes. This can be utilized to detect collisions of the fingers with the environment or in an object placing task to detect contact between object and a supporting surface.

The soft joints of the fingers make joint angle measurement especially challenging as there is no fixed center of rotation and the joint is able to deflect in all three directions of space. While the leaf spring mostly eliminates bending of the joint to the sides, it allows for a considerable amount of twist in the joints. To measure movement in multiple axes, a strong magnet is placed at the joint on the proximal and distal phalanx. On the PCB of the intermediate phalanx, two 3D Hall effect sensors are placed opposite to these magnets. Each combination of joint flexion and twist causes a unique but nonlinear sensor reading that can then be either fitted to an analytical model or alternatively can be calibrated.

In-Hand Vision: In addition to tactile perception, cameras offer an efficient and cheap way of obtaining scene information for grasping and manipulation tasks. Compared to tactile perception, cameras can provide pre-contact information. Cameras can be either installed in a static position in the working setup of the robot, mounted at the robot base or placed in head position. Integrating the camera in the end-effector or artificial hand of a robot comes with the advantage that the view of the contact point between end-effector and scene is visible well in the center of the camera frame. Further, if the hand is regarded as a tool, the tool-center-point is located at a fixed position in the image frame. This configuration is referred to as eye-in-hand visual servoeing and is used for a variety of manipulation tasks where an exact placement of the end-effector is necessary. Depending on the purpose and manipulation strategy, the camera position inside the hand can either be chosen inside the hand palm or inside one of the finger phalanges, as the the following example of the KIT Prosthetic Hand and the KIT Finger-Vision Soft Hand demonstrate.

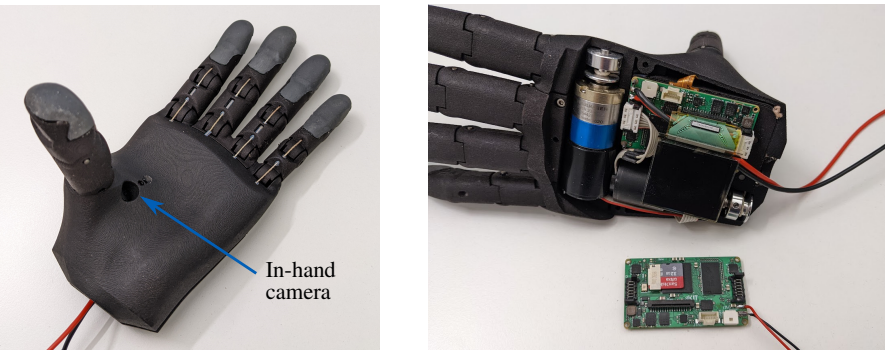

Fig. 16: KIT Prosthetic Hand (Version 2) with in-hand camera in the palm. On the right, the SoC-based embedded in-hand controller board is shown.

When artificial hands are used as prosthetic hands, the controlling of the hand by the user can be quite challenging, since commonly used surface EMG electrodes are limited in resolution and can be unreliable in situations when electrodes are moved or skin properties change. To address this issue, quite a number of researchers have investigated the use of intelligent functions that are based on visual scene perception

to implement semi autonomous control functions [24][23][14]. Visual perception can include shape estimation of objects that are grasped or recognition of specific objects to select from a predefined set of finger trajectories or forces.

The KIT Prosthetic Hand described in [54] is designed to comprehend the scene and thereby suggest a suitable grasp-type to the user. Therefor it has a 1.3 megapixel camera included in the palmar hand side that is used for object recognition. The camera is connected to the hand internal controller, the user can initiate object recognition via a command sent over the EMG interface. For the purpose of image processing and especially image classification tasks, convolutional neural networks are well suited, however they come with a quite high processing workload. Since the hand internal computation resources are very limited due to space and power constraints, special attention was given for a resource aware implementation of the image processing algorithms of the KIT Prosthetic Hand [28]. Since the user expects a grasp suggestion and thereby network result after a certain amount of time, real-time constraints can be derived: The maximum available time slot allows to estimate the highest possible processing workload per network inference. To obtain this value, the available processing time is multiplied with the throughput of the system, meaning the number of operations that the system is capable to perform of each second. To find a network architecture, that matches the available hardware resources and allows the highest accuracy, different optimization techniques can be used as for example genetic algorithms or Bayesian optimization. The processor (Arm Cortex M7) that is used in the first version of the KIT Prosthetic Hand is capable of approximately 2 million multiply-accumulate operations in the given timing budget that are mainly used for processing of convolutional networks. A best suited network architecture was obtained using a genetic algorithm, the network can recognize objects from 13 classed at an accuracy of 96.5%.

The very small size of the commercially available cameras allows to also integrate cameras inside the fingertips of a humanoid hand. This allows to obtain pretouch information from the perspective of the expected contact point. This allows to detect objects, whether they are in reach of the hand, this information can be used in reactive grasping such as handover tasks, grasping of dynamic objects and also precision grasps. The KIT Finger-Vision Soft Hand implements the concept of in-finger vision. Here, each finger includes a miniature camera as shown in Fig. 13. To detect objects that are visible in the camera image, a binary segmentation algorithm is implemented. The segmentation algorithm is realize using a convolutional neural network in an encoder-decoder architecture. The output of the network is a pixel-wise semantic segmentation of all camera images. In a set of experiments the applicability of the concept to a set of typical grasping tasks is shown.

Embedded System: The request for modular and encapsulated hand design comes with the need for a hand embedded controller. The controller is required to fulfill 4 different functions that include reading of sensor values, controlling actuators, processing data and providing an interface to other control units.

To allow processing of data as well as providing interfaces a data processing unit is an essential part of the controller, here the design choice that has to be made is the selection of suitable type of processing system. Commonly used processing

architectures are microcontrollers, application processors, field programmable gate arrays (FPGAs) and System on a chip modules, that can combine logic circuitry and processors. While microcontrollers mostly are used without an operating system and execute relatively simple control algorithms in combination with specialized peripheral modules like times/counters and dedicated interfaces, application processors depend on external memory and an operating system. FPGAs come with the advantage of completely customizable configuration of logic circuits, that allow specialized and parallel interfaces and data processing algorithms. SoCs (system on a chip) can combine processors with reconfigurable logic or other types of hardware accelerators allowing to implement highly efficiency data processing algorithms in combination with complex control methods.

In Fig. 16 the right side image shows a SoC based hand controller for the KIT Prosthetic Hand (50th percentile female). The controller is based on a Xilinx Zynq SoC (XC7Z010) and additional external RAM (2Gb) and 32GB flash storage. The SoC has the advantage of containing a dual core Application processor as well an FPGA. This allows high flexibility for sensor interfaces, allows hardware-accelerated processing of sensor data as well as procedural programming on the processing system. However, the power consumption is comparably high compared to a microcontroller based system. The board includes interfaces for camera, display and relative encoders, as well as two brushed motor drivers. Especially in design of artificial hands, miniaturization of the controller is important, since the mechanisms and actuators already take up large parts of the- available space inside the hand. Therefor, during the mechanical and electrical co-design placement of connectors, routing of cables and heat dissipation must be coordinated carefully. An other important aspect is a robust and easily to assemble and disassemble design, so that in case of any needed modification or mechanical/electrical failure, the system can be accessed and repaired at a reasonable effort.

3.3 Humanoid Head Design

In order for humanoid robots to perceive and interact with their environment, their perception capabilities are a crucial element. Thus, the perception system for such robots should provide sensory information necessary to execute various visuomotor tasks, such as detecting salient regions, and more complex sensorimotor tasks, such as hand-eye coordination, gestures, etc. as well as audio information. For the head, the appearance and the effect on the human observer is especially important, since the face is the center of attention during interactions.

The previously mentioned design principles are also followed in humanoid head design. This applies to modularity and encapsulation of the different components of subsystems of the head. High and robust integration with electrical and mechanical co-design are also very important.

3.3.1 Requirements

The human neck consists of 7 cervical vertebrae which allow an almost continuous bending of the neck in 3 directions The cervical spine can move in flexion (80-90)°, extension 70° (pitch), lateral bending 20-45°(roll), and rotation 90° (yaw) in both directions [49]. This movement is complex since pure rotational movement cannot accurately reproduce it and thus it has to be to be approximated for a robot head that should mimic human neck movements. The peak rotation speed in healthy human is 348 ± 92°/s [43] where the main problem in reaching this speed in robots is acceleration and thus the torque-to-inertia ratio while human muscles have negligible intern inertia. In addition, human eyes allow movements around two major axis in order to be able support gaze control and gaze control.

3.3.2 Kinematics

The kinematics of the human neck can be approximated in a robot by 2 articulation points with 2 DoF each. Various parallel kinematic mechanisms [36] for neck motion are possible, but rarely chosen for humanoid robots. The two 2 DoF of the eyes for direction and gaze stabilization can be realized directly or compensated by camera characteristics like image stabilization. The ability to mimic fast human head movements depends largely on the ratio of head inertia to available drive torque, since acceleration and deceleration phases account for most of the movement.

The kinematic comparison of the robot heads of ARMAR-III, ARMAR-4 and ARMAR-6 show that the first two have a focus on humanoid design and kinematics while the latter is focused on a simple kinematics and the use of certain camera systems. The kinematics of the ARMAR-III head with seven DoF arranged as lower pitch (1), roll (2), yaw (3), upper pitch (4) is shown in Fig. 17a. The complete head with eyes tilt, right eye pan and left eye pan can be seen in 17b. ARMAR-IV has a 9 DOF head with independent eye pan and tilt, a five DOF neck mechanism with lower and upper pitch, lower and upper roll and a yaw joint 17c. In ARMAR-6 and ARMAR-DE a reduced kinematics setup with only pitch and yaw joints was chosen. The order of the joints is pitch-yaw in ARMAR-6 and yaw-pitch in ARMAR-DE (Fig. 17d).

3.3.3 Sensors and Actuation

Accurate position sensing is important for visual perception and hand-eye coordination. The obtained values from the absolute encoders are also used for joint angle control for angular velocity control relative encoders can provide additional feedback in the control loop. Additionally to the actuation of the neck, the cameras can be actuated similarly to the human movable eyes. This enlarges the field of view. ARMAR-III and ARMAR-IV has four digital color cameras each for wide and narrow angle. ARMAR-IV has four cameras and six microphones. The head is also

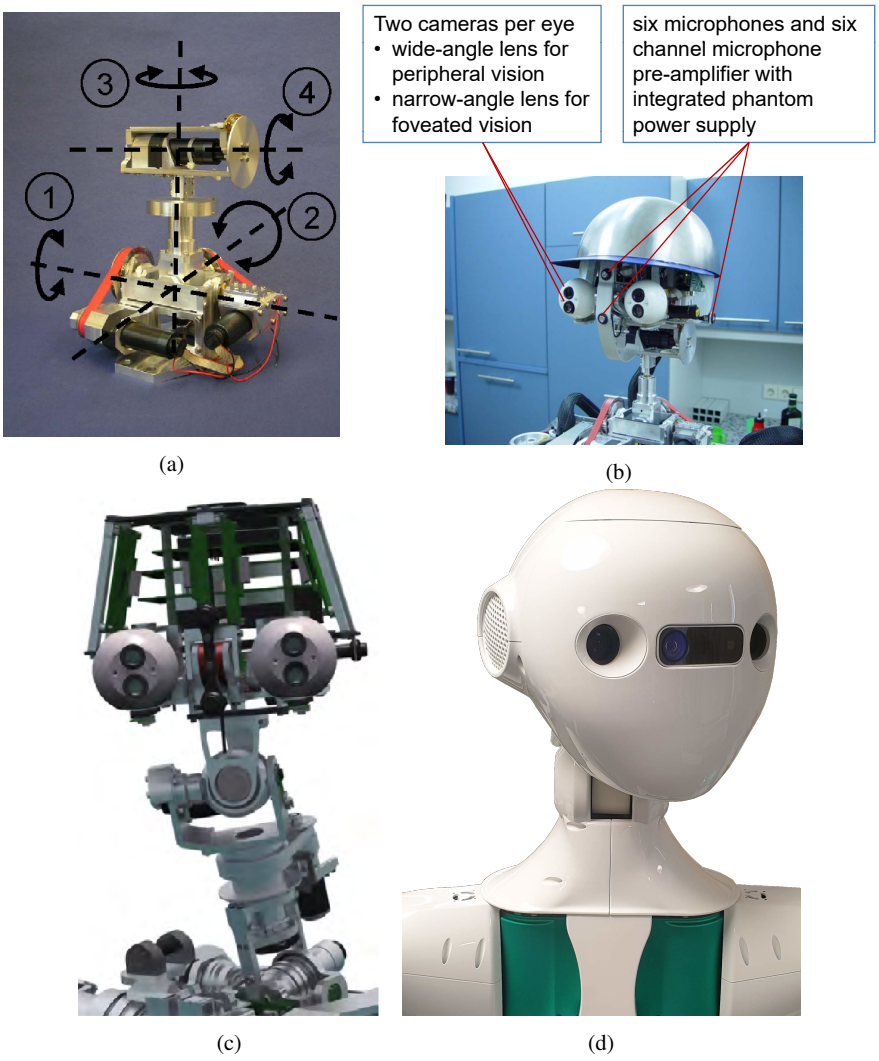

Fig. 17: The neck design of ARMAR-III [1] (a), the head of ARMAR-III [6] with seven DoF (b), the head design of ARMAR-4 [4] (c) and the head of ARMAR-DE (d)

equipped with microphones an an IMU that is used for gaze stabilization and oculomotor control

For actuation speed the possible acceleration is important. This is specified by actuation torques and inertia, where geared motors with typically used high transmission ratios have high internal inertia which is added to the heads inertia. Static forces depend on mass and center of mass. The motors for joint actuation used in

humanoid robots are generally either brushed (DC) or brushless (BLDC) direct current motors. Brushed motors are generally thinner but longer since space for the brushes is needed while BLDC motors are shorter with a larger diameter. The control electronics for BLDC motors is larger and has a minimum size and therefore small motors. Since the torque of small electric motors fitting into the neck and head volume is not sufficient, gearboxes with high transmission ratios are necessary. This can be multi-stage planetary gear boxes or Harmonic Drives which have the advantage of no backlash and where therefore chosen for all actuators in all robot heads. While the joints in ARMAR-6 and ARMAR-DE heads are directly driven, some of the higher number of joints in ARMAR-III and ARMAR-4 heads are driven by belts with remote actuator position.

4 Software Design

Developing software to control complex humanoid robots is a challenging task, that requires a sophisticated software design and architecture. Especially with multiple robots such as ARMAR-III, ARMAR-6, ARMAR-DE, ARMAR-7, re-using functionality is crucial. This can range from pre-processing sensor signals on embedded devices up to autonomous abilities such as grasping an object or navigating through a room. We developed ArmarX [52] as a unified software framework to fulfill these needs. ArmarX emerged over generations of ARMAR robots and is capable of seamlessly and concurrently integrating hardware and software. This allows to maintain and extend existing robots by both its software and hardware as well as its cognitive abilities allowing the integration of new robot components quickly. ArmarX is publicly available under an open-source license[1].

To address the problems of high complexity and transferability, a layered architecture with clear interfaces was designed. The architecture is depicted in Fig. 18 and consists of three layers. 1. The *embedded devices* for highly specialized functionality, 2. the *hardware abstraction layer*, which provides integrating interfaces and abstracts from several bus systems such as USB, Ethernet or EtherCAT, and 3. the *high-level framework* for control and orchestration of multiple robot components which are transferable to other robots. Each of the layers has its individual properties and requirements and will be described in the following sections. One general paradigm that can be found across all layers is the realization of an event-driven closed control cycle as shown in Fig. 19.

For example, on the level of *embedded devices*, there are highly integrated devices with their own firmware-implemented controllers using available relative encoders. On the *hardware abstraction layer*, the SAC units are modeled using a virtual device with two bus participants, namely one motor and one sensing device. Virtual devices also support implementing controllers, using the sensor data of one bus participant and passing control commands to another bus participant. Finally

[1] https://gitlab.com/ArmarX

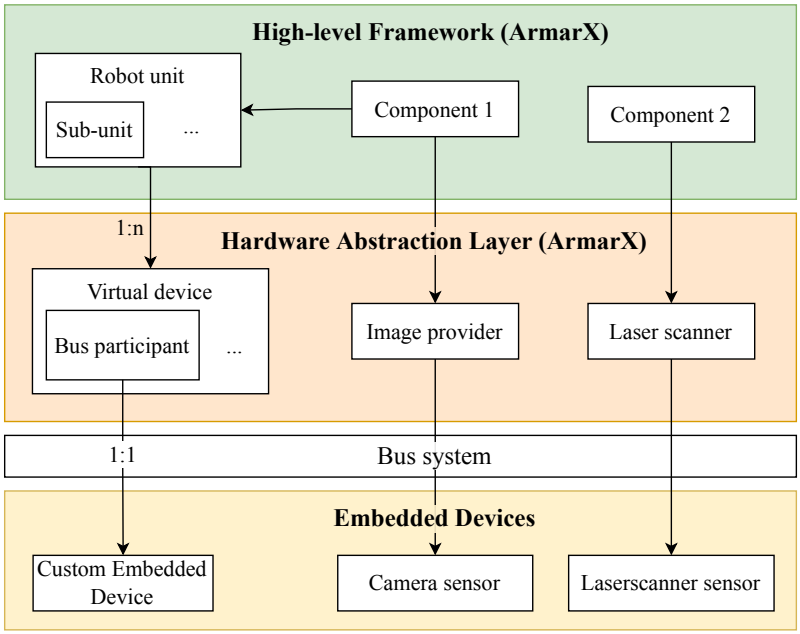

Fig. 18: The three layers of the ArmarX software architecture. **Bottom layer:** Embedded devices, including custom ones, but also commercial sensors such as cameras or laser scanners. **Middle:** Hardware abstraction layer, with unifying interfaces for bus participants (virtual devices, left), cameras (image provider, middle), laser scanners (right), etc. **Top:** High-level framework, with the unifying concept of a robot (robot unit, right), and several exemplary high level components accessing sensors and interacting with the robot unit. Note how several bus systems (e. g., Ethernet, USB, or EtherCAT) are abstracted from using the hardware abstraction layer.

on the *high-level layer*, previously independent components are combined such that

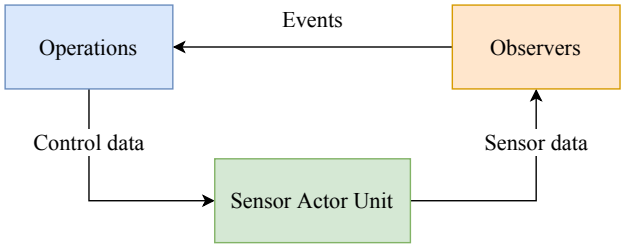

Fig. 19: Event-driven closed control cycle

multi-modal sensor data can be fused to enable e. g., visual servoing or navigation using the laser scanners.

4.1 Embedded Software Design

Embedded software is in most cases rather difficult to change or update once the electronic system is integrated into the robot. The most important design aspects for embedded software in a robot are therefore simplicity and reliability. Mostly the amount of tasks required for an integrated system are clear. They have to communicate with integrated sensor chips such as temperature sensors, absolute or relative encoders, internal measurement units (IMU) or torque sensors, be able to configure them appropriately and periodically and read the current sensor values. Furthermore, they might have to control one or more motors by either directly forwarding PWM values to the motor or by handling position, velocity or torque commands in internal control loops. Finally they have to be able to report if an error happened during the execution of any of their tasks. The last point ensures that failures in robot components can be detected immediately and the robot can be kept in a safe state.

The embedded systems do not only have to communicate with sensors or motors, but need to propagate the read sensor values to a main computer and receive new control commands. For complex robots such as humanoid robots (e. g., ARMAR-6) this communication needs to be fast and reliable due to the vast amount of data that needs to be transferred in every iteration: in ARMAR-6 there are 43 different embedded devices with a total of approximately 3400 B of process data. With an update rate of 1 kHz this results in a required bandwidth of at least 27.2 Mbit/s. Although there are many fieldbus technologies used for different industrial requirements, most of them can not handle this amount of data in such short cycle times. A appropriate fieldbus for this case is e. g., EtherCAT[2], but for smaller robots with less DoF and therefore less periodically data slower fieldbus systems like PROFIBUS[3] or CAN[4] might be sufficient. The low-level interaction on the physical layer and data link layer of the OSI-Model[5]with these fieldbus systems should be handled by either dedicated integrated chips or FPGA systems since the required calculations might use a lot of processing power which is better used for performing the aforementioned tasks of the embedded system.

[2] https://www.ethercat.org/en/technology.html

[3] https://new.siemens.com/global/en/products/automation/industrial-communication/profibus.html

[4] https://www.iso.org/standard/63648.html

[5] https://www.itu.int/rec/T-REC-X.225-199511-I/en

4.1.1 Best Practices for Reliable Embedded Software

In most cases the embedded systems are reasonable simple that there is no need for a real-time operating system (RTOS) but instead the software can be written as one single program which includes all necessary hardware abstractions for on-chip peripherals like UART, SPI, I2C, DMA, timers, GPIO and the drivers for the external peripheral systems like sensors or motors. This program will be executed once the embedded system is powered on and keep running until it gets shutdown by removing the power supply. So-called bare metal programs are today mostly written in C, C++ or Rust but for smaller projects do not require real-time capabilities (Micro)Python is getting more popular[6]. However for absolute control over aspects like memory layout or RAM usage non-interpreted languages are usually advantageous. In the following we provide a few practices for developing embedded software that is based around a single while-loop without a scheduler that handles different threads:

- *No dynamic memory allocations:* Memory should not be allocated during the run time of the program, but instead statically allocated. This has the benefit that the amount of required RAM is known at compile time and therefore prevents possible RAM overflow errors which are impossible to handle at run time.
- *Cooperative multitasking:* The simplest form of a scheduling system. Multiple tasks (e. g., reading values from different sensors) run one after another in a fixed order, but every task is divided into sub tasks, that take minimal execution time. The execution of these sub tasks is handled by a worker function for each task which selects the appropriate sub-task for the current situation. Since every worker always only executes one sub-task, all tasks can run almost in parallel and do not block each other. This principle can be refined by assigning less important tasks a frequency based on the system clock which controls how often the worker does nothing and returns immediately. The programmer has full control over the control flow of the system and does not rely on an underlying scheduler which might introduce non-determinant behavior.
- *Non blocking communication:* Communication over I2C, SPI, UART or other local bus systems should take up as little processing power as possible. It should especially not block other subroutines from being executed, due to for example waiting for a response from an external sensor. The most useful technique here is the usage of DMA (Direct Memory Access) which handles together with dedicated chip hardware the communication completely parallel to the main CPU and only triggers a hardware interrupt once the requested communication step has finished. The programmer can then for example set a certain flag that the data has been successfully received which itself will be handled in the next iteration of the cooperative multitasking loop. Alternatively one could design the PCB itself in a way that more complex communication (e. g., EtherCAT or CAN) gets handled by a specialized integrated chip and the micro controller

[6] https://www.qt.io/embedded-development-talk/embedded-software-programming-languages-pros-cons-and-comparisons-of-popular-languages

only triggers this communication by sending the payload to the chip and gets notified once the communication has finished. This practice goes hand in hand wit cooperative multitasking.

- *Error reporting:* This point is valid for all kind of embedded software. Since the program is running on a system without easy physical access (e. g., because the micro-controller is embedded deep into a mechanical device), it is very difficult to handle malfunctions if there is no way of reporting errors from the embedded system to the main control system. Unlike programs running on a personal computer there is no easy way to debug the system once it is assembled. Important type of errors which are useful to be reported via status words or bit-wise error fields include communication errors with integrated devices like sensors, forwarded error bits from those devices, external signals (like STO) and whether limits of for example temperature sensors are reached.

The design of embedded software depends heavily on the complexity of the underlying embedded system. Smaller robots can be controlled from one large embedded microprocessor, which has the capabilities of running an operating system and can be used and programmed like a small computer. But every robotic system with a high number of DoF needs modular, specialized micro-controller-based embedded systems with programs that need to be as simple and reliable as possible. Some strategies to reach that goal were presented here.

4.2 ArmarX – Hardware Abstraction Layer

The hardware abstraction layer (HAL) is an essential part of the core functionalities of ArmarX and is the foundation all higher components of ArmarX build upon. As shown in Fig. 18, the HAL provides bus-agnostic interfaces for sensors and actors available on the robot to the whole ArmarX ecosystem (potentially spanning across multiple computers inside the robot).

We distinguish two cases for tethering new embedded devices. The simplest case is connecting commercial embedded devices such as cameras or laser scanners via general purpose interfaces (e. g., Ethernet, USB, ...). For these devices, generic software interfaces are already available in ArmarX, e. g., for image providers (cameras), or point cloud providers (laser scanners). The standard procedure, however, is to daisy chain several (custom) embedded devices into a dedicated bus system, in order to save space and to meet hard real-time constraints such as new control targets for actuators. When constructing a robot, we also replicate the physical bus in software by writing device specific software libraries, which integrate embedded devices such as actors or sensors. When starting operation, the physical bus is queried for the bus participants, to find and instantiate the corresponding class describing the state and functionality of each embedded device. These small software libraries for each embedded device allow for a very modular design, as the functionality can be shared between multiple generations of robots, as it is the case with ARMAR-6, ARMAR-DE, and ARMAR-7. New embedded devices can easily be integrated. To

build a new robot, we thus first start by bundling the libraries representing the used embedded devices for a concrete robot.

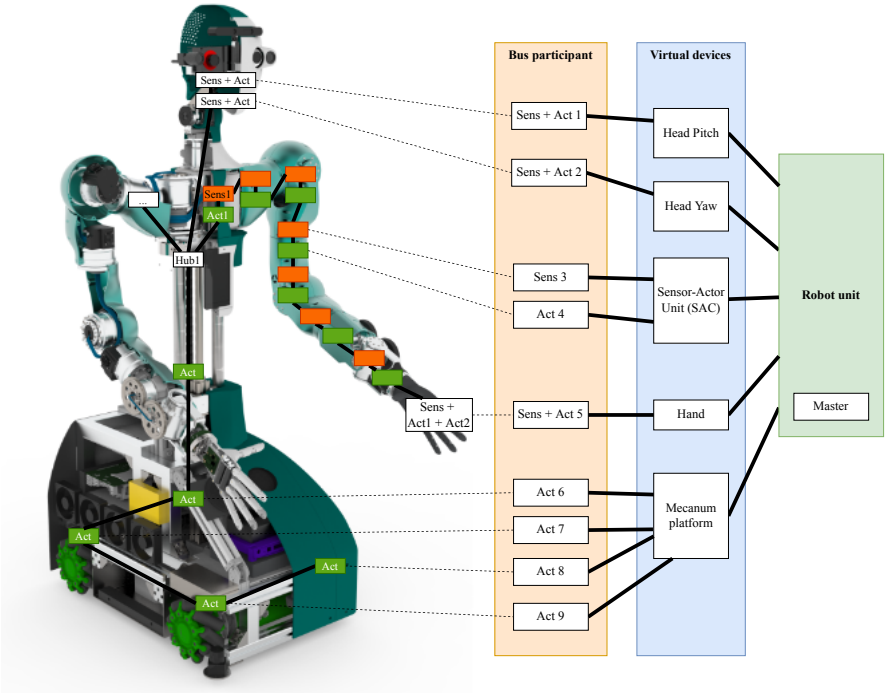

Fig. 20: Schematic of the bus layout in ARMAR-6. Shown is how physical components are mapped to their software counterpart. Virtual devices group one or more bus participants to implement higher-level functionalities. These are then instantiated by the robot unit.

On a higher level of abstraction, we interpret groups of embedded devices on the bus as virtual devices, which are represented in software by objects being assigned to one or more embedded device with the ability to fully control the whole group and thus provide high-level interfaces or high-level control for the remaining framework. An example for this are the wheels of a mobile robotic platform, which are participants of the bus system each on their own. For each wheel (a motor connected to an embedded device on the bus), a software equivalent would be instantiated to provide the low-level functionality in software, such as motor velocity control. On a higher level of abstraction, the mobile platform as a whole is represented through a virtual device in order to provide high-level control by coordinating the set of embedded devices the virtual device was assigned to, e. g., to implement Cartesian platform velocity control by controlling individual wheel velocities.

To ensure *safe operation* of the robot, a continuous monitoring of the system state stops the robot's movement in case of failure. This can be due to software issues e. g., if a crucial component enters a failure state (or crashes) or hardware issues such as the robot being close to self-collision. The implemented state machine is shown in Fig. 21. At startup, the robot enters *safe stop* mode in which the joint

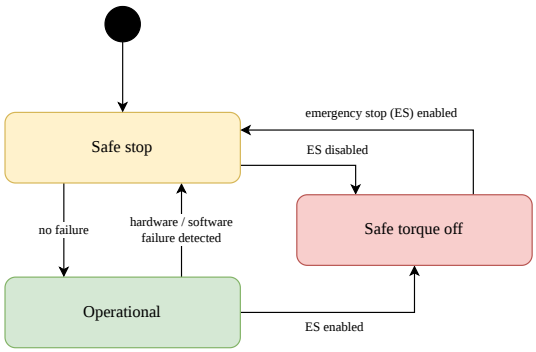

Fig. 21: State machine of the different operational modes. Operational: All control modes are allowed. Safe stop: Joints are zero-velocity controlled, no other control modes allowed. Safe torque off: Joint power supply interrupted, no control possible.

velocities of each joint are controlled to be zero. Entering *operational* mode requires manual intervention of the operator in order to enable the robot to actuate its motors. In case the robot itself detects a software or hardware failure, it returns to safe stop mode. As it might only be a temporary failure, e. g., due to a connection loss to a camera which can be re-established automatically, the robot might recover and transition to operational mode again. In case of unforeseen or unmodeled failures, the operator can also manually bring the robot into safe stop mode. In emergency cases, it is also possible to use any of the hardware emergency stop buttons that are either attached to the robot or connected wirelessly. These directly interrupt the motor's power supply, bringing the robot into *safe torque off* mode. After releasing the hardware emergency stop buttons, the robot will transition into safe stop mode again.

4.3 ArmarX – High-Level Framework

On the high-level layer, robot-specific modules need to be combined and made available in a robot-agnostic way. To do so, all virtual devices that are necessary for the operation of the robot are instantiated by the robot unit. The robot unit also manages the concrete bus as master. As virtual devices might differ between robots, some functionality is exposed via hardware-abstracting sub-unit interfaces that are

common to all robots, e. g., a kinematic unit to control individual joints or a platform unit to move the robot within its local coordinate frame. In addition, the robot unit also manages the real-time controllers for complex task and joint-space control of the whole kinematic chain. These multi-joint controllers are linked to a subset of the kinematic chain. The robot unit ensures that only one controller is active per joint. This allows realizing unimanual, bimanual or even whole-body control. High-level components allow to integrate multi-modal sensor data to build a coherent environment model e. g., by performing vision-based object localization and 6D pose estimation. This enables the realization of complex skills such as vision-based grasping, human-aware navigation and active perception, and many other complex tasks.

4.3.1 Visualization and Introspection

For concurrent hardware and software integration, visualization and introspection tools are essential. For this, the ArmarX GUI can be used to disclose the robot's state. As shown in Fig. 22, the GUI shows information about the robot such as the current joint positions, velocities and torques. In addition, the GUI can be used to control each joint individually. The robots internal state, its location in the environment, as well as the robots model of the environment are visualized on the left side.

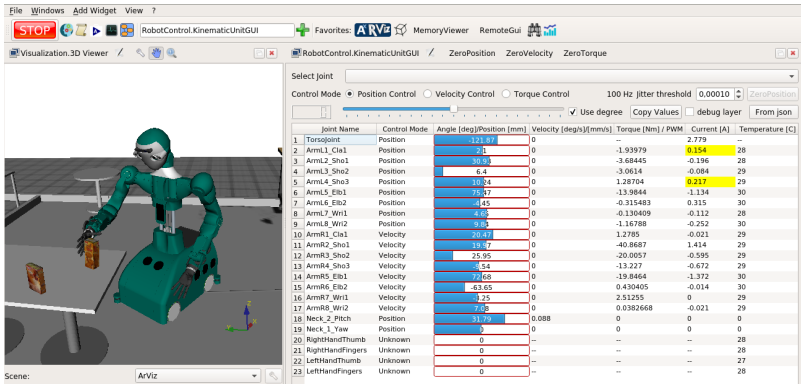

Fig. 22: The ArmarX graphical user interface (GUI). On the left, the virtual scene with ARMAR-6 is shown. The kinematic unit GUI on the right can be used to control all joints of the robot in either position, velocity or torque mode.

5 Conclusions

The development of the mechatronics system for robots is a very complex and time-consuming task. Expert knowledge from the various disciplines of mechatronics is needed to design a robust, highly integrated, high-performance system. Thus, it is important to find solutions to preserve such expert knowledge and made it available for future generation of robot designers. In addition to general guidelines, it would be helpful to provide concrete solutions for the realization of mechatronics components based on previous experience. This requires the formalization of the design process on all levels. To support this process, we developed an ontology-based expert system [32]. Based on user requirements such as spatial dimensions, performance, cost, weight and sensor types, the system proposes concept solutions for the design of mechatronic components. These concept solutions include all required purchased parts, their arrangement to each other and the resulting properties of the overall system. The expert system draws on an ontological knowledge base that preserves the design knowledge from previous developments.

Cross-References

This chapter contains cross-references to content published in other chapters belonging to the four "Robotics Goes MOOC" books; please click on each reference to access the material: KNO2, KNO3, DES5, DES8, INT1, INT2.

Acknowledgements The research leading to the development of the ARMAR humanoid robots has received funding from the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft) within the German Humanoid Research project SFB 588 and the European Union within the the EU projects PACO-PLUS, Xperience, GRASP, WALK-MAN and SecondHands as well as from the German Federal Ministry of Education and Research (BMBF) in the Competence Center ROBDEKON and Innovation Cluster INOPRO. We would like to thank all members of the H^2T lab and all students who contributed to this research in its different phases.

References

1. Albers, A., Brudniok, S., Ottnad, J., Sauter, C., Sedchaicharn, K.: Upper body of a new humanoid robot-the design of armar iii. In: 2006 6th IEEE-RAS International Conference on Humanoid Robots, pp. 308–313. IEEE (2006)
2. Arbib, M.A.: Coordinated control programs for movements of the hand. Experimental Brain Research **10**, 111–129 (1985)
3. Asfour, T., Kaul, L., Wächter, M., Ottenhaus, S., Weiner, P., Rader, S., Grimm, R., Zhou, Y., Grotz, M., Paus, F., et al.: Armar-6: A collaborative humanoid robot for industrial environments. In: 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids), pp. 447–454. IEEE (2018)

4. Asfour, T., Schill, J., Peters, H., Klas, C., Bücker, J., Sander, C., Schulz, S., Kargov, A., Werner, T., Bartenbach, V.: Armar-4: A 63 dof torque controlled humanoid robot. In: 2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 390–396. IEEE (2013)
5. Asfour, T., Waechter, M., Kaul, L., Rader, S., Weiner, P., Ottenhaus, S., Grimm, R., Zhou, Y., Grotz, M., Paus, F.: Armar-6: A high-performance humanoid for human-robot collaboration in real-world scenarios. IEEE Robotics Automation Magazine **26**(4), 108–121 (2019). DOI 10.1109/MRA.2019.2941246
6. Asfour, T., Welke, K., Azad, P., Ude, A., Dillmann, R.: The karlsruhe humanoid head. Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots pp. 447–453 (2008)
7. Barry, B., et al.: Software engineering economics. New York **197** (1981)
8. Belter, J.T., Dollar, A.M.: Novel Differential Mechanism Enabling Two DOF from a Single Actuator: Application to a Prosthetic Hand. In: IEEE International Conference on Rehabilitation Robotics (ICORR), pp. 1–6. Seattle (2013). DOI 10.1109/ICORR.2013.6650441
9. Boehm, B.W.: A spiral model of software development and enhancement. Computer **21**(5), 61–72 (1988)
10. Brown, C., Asada, H.: Inter-Finger Coordination and Postural Synergies in Robot Hands Via Mechanical Implementation of Principal Components Analysis. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2877–2882 (2007)
11. Carryer, J., Ohline, R., Kenny, T.: Introduction to Mechatronic Design. Prentice Hall (2011)
12. Catalano, M., Grioli, G., Farnioli, E., Serio, A., Piazza, C., Bicchi, A.: Adaptive Synergies for the Design and Control of the Pisa/IIT SoftHand. Int. Journal of Robotics Research **33**(5), 768–782 (2014). DOI 10.1177/0278364913518998
13. Chen, X., Zhang, X., Huang, Y., Cao, L., Liu, J.: A review of soft manipulator research, applications, and opportunities. Journal of Field Robotics **n/a**(n/a) (2021). DOI 10.1002/rob.22051
14. DeGol, J., Akhtar, A., Manja, B., Bretl, T.: Automatic grasp selection using a camera in a hand prosthesis. In: 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 431–434. IEEE (2016)
15. Deimel, R., Brock, O.: A novel type of compliant and underactuated robotic hand for dexterous grasping. Int. Journal of Robotics Research **35**(1-3), 161–185 (2016)
16. Dollar, A.M., Howe, R.D.: Joint coupling design of underactuated hands for unstructured environments. The International Journal of Robotics Research **30**(9), 1157–1169 (2011). DOI 10.1177/0278364911401441
17. Forsberg, K., Co-Principals, H.M.: 4 system engineering for faster, cheaper, better. In: INCOSE International Symposium, vol. 9, pp. 924–932. Wiley Online Library (1999)
18. Fukaya, N., Asfour, T., Dillmann, R., Toyama, S.: Development of a Five-Finger Dexterous Hand without Feedback Control: The TUAT/Karlsruhe Humanoid Hand. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Tokyo, Japan (2013)
19. Fukaya, N., Toyama, S., Asfour, T., Dillmann, R.: Design of the TUAT/Karlsruhe Humanoid Hand. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Takamatsu, Japan (2000)
20. Gaiser, I., Schulz, S., Kargov, A., Klosek, H., Bierbaum, A., Pylatiuk, C., Oberle, R., Werner, T., Asfour, T., Bretthauer, G., Dillmann, R.: A new anthropomorphic robotic hand. In: Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots, pp. 418–422 (2008). DOI 10.1109/ICHR.2008.4755987
21. Gaiser, I., Wiegand, R., Ivlev, O., Andres, A., Breitwieser, H., Schulz, S., Bretthauer, G.: Compliant Robotics and Automation with Flexible Fluidic Actuators and Inflatable Structures. IntechOpen (2012)
22. Gausemeier, J., Moehringer, S.: New guideline vdi 2206-a flexible procedure model for the design of mechatronic systems. In: DS 31: Proceedings of ICED 03, the 14th International Conference on Engineering Design, Stockholm (2003)
23. Ghazaei, G., Alameer, A., Degenaar, P., Morgan, G., Nazarpour, K.: Deep learning-based artificial vision for grasp classification in myoelectric hands. Journal of neural engineering **14**(3), 036,025 (2017)

24. Giordaniello, F., Cognolato, M., Graziani, M., Gijsberts, A., Gregori, V., Saetta, G., Hager, A.G.M., Tiengo, C., Bassetto, F., Brugger, P., et al.: Megane pro: myo-electricity, visual and gaze tracking data acquisitions to improve hand prosthetics. In: 2017 International Conference on Rehabilitation Robotics (ICORR), pp. 1148–1153. IEEE (2017)
25. Grebenstein, M., Albu-Schäffer, A., Bahls, T., Chalon, M., Eiberger, O., Friedl, W., Gruber, R., Haddadin, S., Hagn, U., Haslinger, R., Höppner, H., Jörg, S., Nickl, M., Nothhelfer, A., Petit, F., Reill, J., Seitz, N., Wimböck, T., Wolf, S., Wüsthoff, T., Hirzinger, G.: The DLR hand arm system. In: 2011 IEEE International Conference on Robotics and Automation, pp. 3175–3182 (2011)
26. Grebenstein, M., Chalon, M., Friedl, W., Haddadin, S., Wimböck, T., Hirzinger, G., Siegwart, R.: The hand of the DLR Hand Arm System: Designed for interaction. The International Journal of Robotics Research **31**(13), 1531–1555 (2012). DOI 10.1177/0278364912459209
27. Hundhausen, F., Grimm, R., Stieber, L., Asfour, T.: Fast reactive grasping with in-finger vision and in-hand fpga-accelerated cnns. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 0–0 (2021)
28. Hundhausen, F., Megerle, D., Asfour, T.: Resource-aware object classification and segmentation for semi-autonomous grasping with prosthetic hands. In: 2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids), pp. 215–221. IEEE (2019)
29. Hundhausen, F., Starke, J., Asfour, T.: A soft humanoid hand with in-finger visual perception. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 8722–8728 (2020)
30. Kappassov, Z., Corrales, J.A., Perdereau, V.: Tactile sensing in dexterous robot hands — review. Robotics and Autonomous Systems **74**(1) (2015). DOI 10.1016/j.robot.2015.07.015
31. Kargov, A., Werner, T., Pylatiuk, C., Schulz, S.: Development of a miniaturised hydraulic actuation system for artificial hands. Sensors and Actuators A: Physical **141**(2), 548–557 (2008). DOI https://doi.org/10.1016/j.sna.2007.10.025
32. Karrenbauer, O., Rader, S., Asfour, T.: An ontology-based expert system to support the design of humanoid robot components. In: 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids), pp. 1–8. IEEE (2018)
33. Kyberd, P.J., Chappell, P.H.: A force sensor for automatic manipulation based on the hall effect. Measurement Science and Technology **4**(3), 281–287 (1993). DOI 10.1088/0957-0233/4/3/005
34. Laliberte, T., Birglen, L., Gosselin, C.: Underactuation in robotic grasping hands. Machine Intelligence & Robotic Control **4**(3), 1–11 (2002)
35. Landsmeer, J.M.F.: TheCoordination of Finger-Joint Motions. The Journal of Bone & Joint Surgery **45**(8), 1654–1662 (1963)
36. Merlet, J.P., Gosselin, C., Huang, T.: Parallel mechanisms. In: Springer handbook of robotics, pp. 443–462. Springer (2016)
37. Mohd Faudzi, A.A., Ooga, J., Goto, T., Takeichi, M., Suzumori, K.: Index Finger of a Human-Like Robotic Hand Using Thin Soft Muscles. IEEE Robotics and Automation Letters **3**(1), 92–99 (2018)
38. Odhner, L.U., Jentoft, L.P., Claffee, M.R., Corson, N., Tenzer, Y., Ma, R.R., Buehler, M., Kohout, R., D. Howe, R., Dollar, A.M.: A compliant, underactuated hand for robust manipulation. The International Journal of Robotics Research **33**(5) (2014). DOI 10.1177/0278364913514466
39. Park, M., Bok, B.G., Ahn, J.H., Kim, M.S.: Recent advances in tactile sensing technology. Micromachines **9**(7) (2018). DOI 10.3390/mi9070321
40. Patel, R., Cox, R., Correll, N.: Integrated proximity, contact and force sensing using elastomer-embedded commodity proximity sensors. Autonomous Robots **42**(7), 1443–1458 (2018). DOI 10.1007/s10514-018-9751-4
41. Piazza, C., Grioli, G., Catalano, M., Bicchi, A.: A century of robotic hands. Annual Review of Control, Robotics, and Autonomous Systems **2**(1), 1–32 (2019). DOI 10.1146/annurev-control-060117-105003
42. Rader, S., Kaul, L., Weiner, P., Asfour, T.: Highly integrated sensor-actuator-controller units for modular robot design pp. 1160–1166 (2017). DOI 10.1109/AIM.2017.8014175

43. Röijezon, U., Djupsjöbacka, M., Björklund, M., Häger-Ross, C., Grip, H., Liebermann, D.G.: Kinematics of fast cervical rotations in persons with chronic neck pain: a cross-sectional and reliability study. BMC musculoskeletal disorders **11**(1), 1–10 (2010)
44. Salem, F.A., Mahfouz, A.A.: Mechatronics design and implementation education-oriented methodology; a proposed approach. Mechatronics **1**(3) (2014)
45. Santello, M., Flanders, M., Soechting, J.F.: Postural Hand Synergies for Tool Use. The Journal of Neuroscience **18**(23), 10,105–10,115 (1998). DOI citeulike-article-id:423192
46. Saudabayev, A., Varol, H.A.: Sensors for robotic hands: A survey of state of the art. IEEE Access **3**(0) (2015). DOI 10.1109/ACCESS.2015.2482543
47. Schulte, H.F.: The characteristics of the McKibben artificial muscle. In: National Academy of Sciences - National Research Council (1961)
48. Smit, G., Plettenburg, D.H., van der Helm, F.C.T.: The Lightweight Delft Cylinder Hand: First Multi-Articulating Hand That Meets the Basic User Requirements. IEEE Transactions on Neural Systems and Rehabilitation Engineering **23**(3), 431–440 (2015)
49. Swartz, E.E., Floyd, R., Cendoma, M.: Cervical spine functional anatomy and the biomechanics of injury due to compressive loading. Journal of athletic training **40**(3), 155 (2005)
50. Tenzer, Y., Jentoft, L.P., Howe, R.D.: The feel of mems barometers: Inexpensive and easily customized tactile array sensors. IEEE Robotics Automation Magazine **21**(3) (2014). DOI 10.1109/MRA.2014.2310152
51. Vahrenkamp, N., Asfour, T., Dillmann, R.: Efficient inverse kinematics computation based on reachability analysis. International Journal of Humanoid Robotics (IJHR) **9**(4), 0–0 (2012)
52. Vahrenkamp, N., Wächter, M., Kröhnert, M., Welke, K., Asfour, T.: The Robot Software Framework ArmarX. it – Information Technology **57**(2), 99–111 (2015). DOI 10.1515/itit-2014-1066
53. Weiner, P., Hundhausen, F., Grimm, R., Asfour, T.: Detecting grasp phases and adaption of object-hand interaction forces of a soft humanoid hand based on tactile feedback. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3956–3963 (2021)
54. Weiner, P., Starke, J., Hundhausen, F., Beil, J., Asfour, T.: The KIT Prosthetic Hand: Design and Control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3328–3334. Madrid, Spain (2018)
55. Weiner, P., Starke, J., Rader, S., Hundhausen, F., Asfour, T.: Designing prosthetic hands with embodied intelligence: The kit prosthetic hands. Frontiers in Neurorobotics (2022). (Currently under review)
56. Werner, T., Kargov, A., Gaiser, I., Bierbaum, A., Schill, J., Schulz, S., Bretthauer, G.: Eine fluidisch angetriebene anthropomorphe roboterhand. at - Automatisierungstechnik **58**(12), 681–687 (2010). DOI 10.1524/auto.2010.0877
57. Yamaguchi, N., Hasegawa, S., Okada, K., Inaba, M.: A gripper for object search and grasp through proximity sensing. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1–9 (2018). DOI 10.1109/IROS.2018.8593572
58. Zheng, C., Bricogne, M., Le Duigou, J., Eynard, B.: Survey on mechatronic engineering: A focus on design methods and product models. Advanced Engineering Informatics **28**(3), 241–257 (2014). DOI 10.1016/j.aei.2014.05.003. Multiview Modeling for Mechatronic Design