Virtual Sensors: A Comprehensive Review on Robotic Applications

Patrick Kosierb[a], Thomas French[a], Alex McCafferty-Leroux[a], Yuandi Wu[a], S. Andrew Gadsden[a]

[a]*Mechanical Engineering McMaster University, 1280 Main St. W, Hamilton, L8S4L8, ON, Canada*

Abstract

Many challenges are associated with modern robotic manipulators, such as ensuring redundancy, reliability, and adaptability. One challenge in particular is the operation of robotic manipulators in the harsh environment of space. Another lies in maintaining safety in the ever-evolving field of collaborative robotics, where collisions are prone to excessive liability. In addition, further advancements in simulation software, computational power, machine learning approaches, and emerging trends of cyber-physical systems are increasing data demands to improve control and safety systems. Another prevailing challenge is the affordability of industrial-grade robotics due to the intricate sensors and actuators onboard. Virtual sensors can supplement these demands and tackle current challenges by producing accurate and highly correlated measurements for otherwise expensive or degraded physical sensors. This paper provides a comprehensive survey of current trends in virtual sensors applied to robotic manipulators. The motivation is identifying highly applicable strategies and gaps in virtual sensor architectures to further develop the field.

Keywords: virtual sensors, soft sensors, force/torque sensor, friction, robotics.

1. Introduction

Robotic manipulators, or robot arms, continue to revolutionize various industries. An outstanding accomplishment among robotics engineers and researchers is the marvellous feats of brilliance seen in the space sector. To

highlight one key player, Canada's Space Station Remote Manipulator System (SSRMS), also known as Canadarm2, is a multipurpose teleoperated and semi-autonomous manipulator stationed on the International Space Station (ISS). A few substantial achievements are spacecraft servicing, in-space construction, extra vehicular activity (EVA) support, and target capture [**?**]. Apart from the zero-g benefits attributed to space mechanics, the environment in space is harsh for robotics. The composition of mechanical components and avionics that form the robot arm must be able to withstand extreme temperatures as low as $-100°C$ and as high as $100°C$. At extreme temperatures, sensitive onboard instruments such as force/torque sensors experience drift due to thermal expansion of materials, which can be reflected in strain gauge measurements if thermal management is not handled effectively. Vast amounts of electromagnetic interference (EMI) caused by solar flares and other cosmic rays threaten delicate electronic components by inducing currents, consequently, leading to component breakdown [**?**]. Another complication caused by EMI is hindering signal integrity (i.e. creating noise), which means that data coming from sensors onboard the robot may not reflect true measurements. These obstacles already have solutions in place, for example, cyclic-redundancy checks (CRC) are done in firmware to ensure bit flips are not occurring among digital electronic components such as flip-flops, and reliable differential-signal protocols like controller area network (CAN) are used to minimize the mentioned noise. However, faults always occur. Erring on caution by maintaining redundancies and refining safety measures through technological advancements is necessary and paramount for future space robotics.

Further advancements in manufacturing processes under the Industry 4.0 storm, such as additive manufacturing and advanced computer-aided manufacturing (CAM) techniques, have established a reliance on robotic arms to meet such sophisticated plans [**?**]. A challenge associated with robotic arms in manufacturing is affordability and collaboration with human workers. One such collaborative robot (cobot) is Universal Robotics UR10, a large 6-degree-of-freedom (dof) robot that can work hand in hand with humans safely. To become a cobot, rigorous safety procedures and guidelines must be followed (e.g. max joint speed, safety stops, boundary conditions) to ensure that the robot is not a liability in the workplace [**?**]. However, emerging technologies involving human-robot interaction, such as online learning [**?**] and perception capabilities [**?**], can benefit from additional safety measures and redundancies regarding contact and collision forces that heavily rely on

physical measurements. Typically, the hardware architecture for a robotic arm is complex, with many sensors on board that collect data for key parameters such as joint angles, velocities, motor temperature, and forces applied to the end effector. The demands above can be supplemented with more onboard sensors, but drawbacks exist. Some include increased costs, especially for high-precision sensors used at the end effector, weight constraints, the inability to install a physical sensor due to space constraints, and increased power consumption. For example, force-torque sensors typically installed at the end-effector or base of the robot arm are costly due to the manufacturing process required to yield high accuracy. In addition, calibration is required every so often for accurate and reliable measurements, and as previously mentioned, relies on a baseline assumption of environmental conditions. In some cases, the calibration process must be done by laboratories that meet standards such as ISO 1702, which further ramps up costs [**?**].

Virtual sensors, also known as soft sensors, can play a role in satisfying these demands while proving advantageous over physical sensors, overcoming the drawbacks mentioned. Virtual sensors are software constructs that produce signals like a physical sensor, except they are not limited to quantities that are measurable by a sensor. A virtual sensor can make measurements for parameters that do not have a physical counterpart using a combination of real measurements, estimation theory, mathematical relationships, and machine learning techniques [**?**]. For example, the internal temperature of windings in a brushless direct current (DC) motor cannot be intuitively measured via a physical sensor without potentially disrupting crucial motor parameters such as phase resistance. With this in mind, a virtual sensor can estimate the internal winding temperature via other measurements, motor properties, and environmental conditions. Algorithms that may comprise a virtual sensor have already been applied to robotic arms, such as Kalman filtering (KF) techniques, machine learning for system identification, and various forms of computer vision for environmental awareness. However, the synergy of these methods in a novel way can lead to explicit optimizations such as removing physical sensors and replacing them with a virtual counterpart, becoming reliable redundancies for physical sensors, or determining variables that cannot be directly measured with a physical sensor.

The exact definition of a virtual sensor is ambiguous, as it's an emerging topic with many similarities and overlaps with well-defined estimation methods. The remainder of this section will outline virtual sensing found in the literature. A comprehensive review of data-driven virtual sensors for

water quality parameters is given in [**?**], including architectures followed for virtual sensor development. A survey on virtual sensor networks that utilize wireless networks for deploying services for other nodes on the network can be seen in [**?**], along with a proposed architecture for the network and possible applications, including battlefield surveillance, vehicle monitoring, and smart home monitoring. Data-driven soft sensors, specifically using transfer learning, are surveyed in [**?**]. A review of methods for developing regression-based soft sensors is given in [**?**]. A summary of potential deep learning models for developing soft sensors is given in [**?**], along with reviewing current soft sensors in the literature. Input selection methods for virtual sensors used for industrial processes are evaluated in [**?**] to optimize the virtual sensing models' performance and computation cost. Challenges and trends with virtual sensing methods used in industrial processes are reviewed in [**?**], specifically addressing how they react to process changes, differences in equipment, and reliability.

There is a lack of literature regarding the application of virtual sensors in robotic systems, specifically robotic manipulators. This survey will define a virtual sensor, outline its application for robotic arms, and identify trends and gaps relating to its deployment. The remainder of the paper will be structured as follows, Section 2 will outline preliminaries on virtual sensors and the internal methods used for virtual sensor architectures, Section 3 will survey the application of virtual force/torque sensors in robotic arms, Section 4 highlights virtual sensors for uncertainties and friction in robotic arms, Section 5 is the discussion on virtual sensors in robotics, and Section 6 is the conclusion.

2. Virtual Sensor Preliminaries

At the highest level of abstraction, a virtual sensor is a mathematical model unique to a given application that takes various forms of telemetry from the system as input and outputs a desired parameter. Further, a virtual sensor can be broken down into three components. The first is the system or parameter being measured. Second is the system's data or telemetry (i.e. physical sensors, virtual sensors, input, state variable, etc.). Third, a merging function that transforms real sensory data or other system information into a virtual sensor signal. There can be various configurations for the input to a virtual sensor, for example, a single physical sensor, multiple physical sensors, or a combination of physical and virtual sensors [**?**]. See Figure 1

for a high-level architectural diagram. The following section will outline the data fusion methods used for virtual sensors.

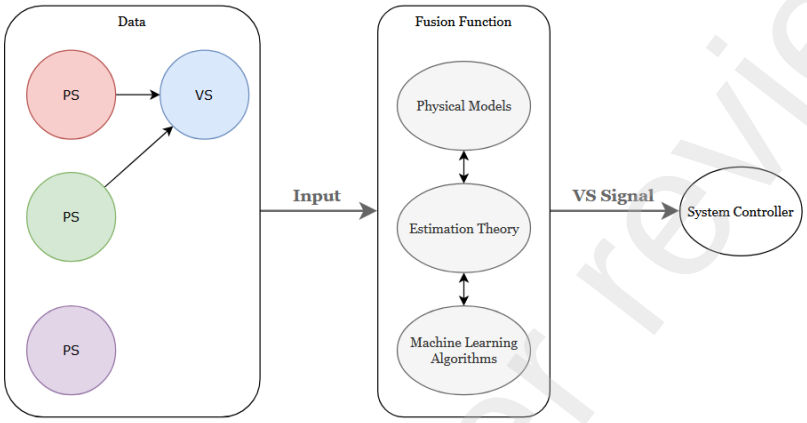

Figure 1: Virtual sensor architecture with three physical sensors (PS) and one virtual sensor (VS) as input and a combination of physical models, estimation, and machine learning algorithms as the virtual sensor function. The output is sent to the system controller.

2.1. Virtual Sensor Function

The virtual sensor function is the core of the virtual sensor, creating a desired measurement by manipulating a unique combination of information from the system. There are three architectures a virtual sensor function can follow: (1) using a physical model, (2) knowledge-based/estimation, (3) data-driven approach [**?**]. These can be classified as white-box, grey-box, and black-box approaches, respectively [**?**]. It's important to note that a merging function can be a combination of the three approaches. The virtual sensor function can be defined as

$$y_{vs}^{i} = \nu(y_{ps}, y_{vs}, x, u), \tag{1}$$

where y_{vs} is a vector of virtual sensor measurement (i being the index for the desired measurement), ν is the fusion function, y_{ps} is a vector of raw physical sensor measurements, x is the state vector of the system, and u is the control input vector. A virtual sensor with virtual sensors as input can be defined as a *nested virtual sensor*.

2.2. White-box approach

The white-box approach is typically a reliable mathematical model of the measured parameter that involves some form of approximation to the real true value due to pre-defined assumptions. For example, in the context of robotics, a white-box approach can be the relationship between the inverse kinematics or dynamics to derive a specific joint or Cartesian space parameter or the relationship between temperature, torque, and current for a DC or permanent magnet alternating current (PMAC) motor. If one desires measurements of the internal joint friction, a mathematical relationship representing the Coulomb and viscous friction may be used as the virtual sensor construct. To further advance this method, the soft sensor can be paired with a black-box approach for recording uncertainties in the friction models.

2.3. Grey-box approach

The grey-box approach encapsulates data acquisition with state estimation, given knowledge about a stochastic system. For example, applying Bayesian filtering algorithms on noisy measurement data for state estimation, where the states are used as input to the virtual sensing algorithm. A Kalman filter (KF) can be applied to linear systems, yielding the optimal state estimate by minimizing the state error covariance leading to a predictable algebraic solution. Non-linearities introduce uncertainty because an approximation must be made to maintain the Gaussian noise assumption. Methods like the unscented Kalman filter (UKF) account for non-linearities by utilizing the unscented transform to approximate the non-linear distribution with a normal distribution [**?**]. Although stochastic processes are involved, the underlying methods for state estimation are deterministic, assuming consistent initial parameters, process, and measurement noise matrices, hence the grey-box classification [**?**]. There are other estimation approaches to consider, such as the particle filter, histogram filter, and the H-infinity filter.

2.4. Black-box approach

The black-box method derives a virtual sensor using a heuristic approach where the output is non-deterministic. The unexplanability of black-box approaches poses a challenge for safety-critical applications such as the space sector; however, explainable AI and a human in the loop can mitigate these concerns. The use of deep learning algorithms can be deemed black-box since the derivation of the model is largely up to the training procedures,

hyper-parameter optimization, and the initial conditions where many possible solutions can fit a particular application even if not the global minimum for the loss function. A virtual sensor can use a multi-modal artificial neural network (ANN) that takes in a variety of real measurements with a single output neuron with the motivation of modelling a highly non-linear relationship within the system that cannot be achieved with a deterministic method [**?**].

2.5. Virtual Sensor example

An example virtual sensor can be formulated for measuring the reaction force of a 1-dof fluid damper. In this case, the formulation will be a combination of the three approaches mentioned. The basis for the virtual sensor is the mathematical model used to represent the damping force. System identification must be completed to solve for unknown parameters in this model, and an estimation approach can be used to improve the damping force output. The three methods used will encompass the virtual sensor.

Going into more detail, a magnetorheological damper can work as the 1-dof system in question, and the mathematical model used to represent the damping force can be the Bouc-Wen model [**?**]. System identification can be done with a generic algorithm (GA), and further refinement can occur as the system is active. In a real-time setting, a UKF can be used with an augmented state vector [**?**] for force estimation using the predefined model found with the GA. The input to the virtual sensor is the command voltage and last state vector, along with the hyperparameters required for the GA and UKF. The output of the virtual sensor is the damping force, see Figure 2 for a high-level diagram of the virtual sensor.

3. External Contact Virtual Sensors

A prominent research topic in robotics is sensing external contacts, both the position of the contact relative to a frame and the magnitude of the wrench. To perform tasks with high precision, having confidence in the wrench feedback is important; in addition, sensing external contacts (i.e. wrench) is crucial for safety in a collaborative robot environment to detect unwanted collisions with factory equipment or workers. In industry, it's common for a robotic manipulator to be equipped with a 6-axis force/torque sensor at the tool tip for precise force control when performing various manufacturing tasks. However, these sensors tend to be expensive relative to the

Table 1: Literature for virtual sensing on robotic arms (sorted by year).

Author	Source	Year	Type	Output
Kennedy and Desai	[?]	2003	Model-based	Friction Torque
Petrovskaya et al.	[?]	2007	NLLS + Baye's Theorem	Contact points and geometry
Magrini et al.	[?]	2014	FOMO + Computer vision	External force/torque and contact point
Del Sol et al.	[?]	2014	Model-based	External force/torque
Buondonno and De Luca.	[?]	2016	FOMO + RNEA	External force/torque and contact point
Berger et al.	[?]	2016	Feature space projections	External force/torque and contact point
Sun et al.	[?]	2016	PSO+LSSVM	Uncertainty torque
Villagrossi et al.	[?]	2018	Model-based + GA	External force/torque and Friction force/torque
Mancisidor et al.	[?]	2018	Model-based	External force/torque and contact point
Wahrburg et al.	[?]	2018	KF vs. FOMO	External force/torque
Hu and Xiong.	[?]	2018	DKF	External force/torque
Yen et al.	[?]	2019	FOMO	External force/torque
Li et al.	[?]	2019	FOMO	External force/torque and collision identification
Gold et al.	[?]	2019	FOMO + Sensor fusion	External force/torque
Garofalo et al.	[?]	2019	FOMO vs SOSM vs SOSML	External force/torque
Shin et al.	[?]	2019	Model-based	External force/torque and pressure
Lim et al.	[?]	2021	FOMO + LSTM	External force/torque
Yigit et al.	[?]	2021	FOMO vs EKF vs ANN	External force/torque
Cao et al.	[?]	2021	AKF vs KF	External force/torque
Pang et al.	[?]	2021	RSGD	External force/torque and contact point
Hung and Jiang.	[?]	2023	Model-based + XGBoost	External force/torque
Jiang et al.	[?]	2023	FOMO vs SOMO vs ISOMO	External force/torque
Sorrentino et al.	[?]	2024	UKF vs RNEA	External force/torque

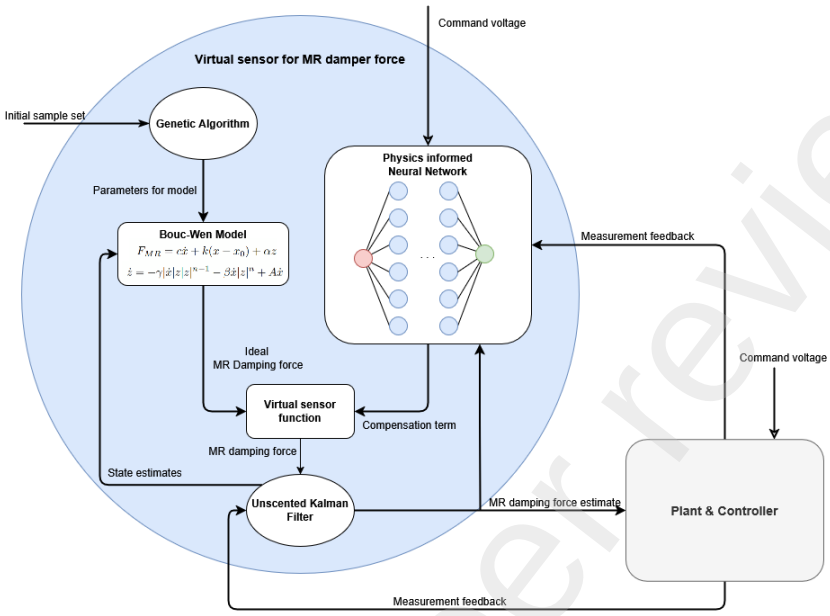

Figure 2: Example virtual sensor architecture for MR damper force.

rest of the robot [**?**] and experience substantial drift in non-ideal environments [**?**]. In addition, using virtual sensors for external contact estimation introduces a layer of redundancy, eliminating a single point of failure in the force control domain which is especially important in high-risk missions such as space maintenance. Therefore, it's highly desirable to measure external contacts without using a physical sensor or including a construct to validate measurements from the physical sensor. This section will outline potential external contact virtual sensors for robotic manipulators found in the literature, some of which include deriving contact point positions. An insightful survey on a similar topic includes [**?**], where robot collision detection methods are highlighted.

3.1. Model-based

3.1.1. Direct Dynamics

The obvious approach to external wrench derivation is through direct use of the general dynamics formulation for robotic manipulators. The following equation represents the general dynamic formulation derived from the Lagrangian of an N-dof robotic manipulator

$$\tau_c = M(q)\ddot{q} + h(q, \dot{q}) + \tau_{ext} + \tau_f, \tag{2}$$

where $\tau_c \in \mathbb{R}^N$ is joint control torque, $M(q) \in \mathbb{R}^{N\times N}$ is the mass matrix that depends on the rigid body properties, $q \in \mathbb{R}^N$ is the joint angular position vector, $\dot{q} \in \mathbb{R}^N$ is the angular velocity vector, $\ddot{q} \in \mathbb{R}^N$ is the angular acceleration vector, $\tau_{ext} \in \mathbb{R}^N$ is the joint external torque generated by a contact force, $\tau_f \in \mathbb{R}^N$ is joint friction torque, and $h(q, \dot{q}) \in \mathbb{R}^N$ is a term that is used to lump torques caused by Coriolis, centrifugal, and gravitational effects together

$$h(q, \dot{q}) = C(q, \dot{q})\dot{q} + g(q), \tag{3}$$

where $C(q, \dot{q}) \in \mathbb{R}^{N\times N}$ is the Coriolis and centrifugal effect matrix and $g(q) \in \mathbb{R}^N$ is the torque produced by gravity. To differentiate between contact tool center point (TCP) force and torque, the following relationship can be used

$$\tau_{ext} = J^T(q)\mathcal{F}, \tag{4}$$

where $J \in \mathbb{R}^{N\times N}$ is the geometric Jacobian matrix, and $\mathcal{F} \in \mathbb{R}^6$ is the TCP force and moments (external wrench) in the Cartesian frame defined as

$$\mathcal{F} = \begin{bmatrix} F_{tcp} \\ M_{tcp} \end{bmatrix} = [F_x, F_y, F_z, M_x, M_y, M_z]^T. \tag{5}$$

It should be noted that more often than not, part of $\mathcal{F}$ is usually zero depending on the nature of the contact/collision [**?**]. Also, an additional note on notation, for any variable (a), a *hat* will be used as its corresponding estimate $(\hat{a})$.

Although this method is straightforward and provides an analytical approach to solving external torques applied to a robot when assuming any unaccounted-for torques to an external contact, there is a downside. The angular acceleration used in the general dynamics equation is not a directly measured quantity in most robot arms [**?**]. To find this term, a double differentiation of encoder or resolver data would need to be done, which amplifies noise in the measurements. Further inaccuracies with relying on the general dynamics form can stem from poor definitions of rigid body parameters (inertia tensors, center of mass, etc.) found in the dynamic matrices. To justify the use of the dynamics equation in safety-critical industries such as the space sector, a sensitivity analysis can be conducted to verify inertial

tensors. Command torque can be modelled with the fundamental equations for the current-torque relationship in motors. For any joint i

$$\tau_c^i = GK_I I_c^i, \tag{6}$$

where G is the transmission ratio (gearbox dependent), K_I (typically $K_I^1 \approx K_I^2 \approx \cdots \approx K_I^i = K_I$ if motors are alike) is the torque constant, and I_c^i, is the applied motor current. The assumption is that the command torque is well known; in reality, gearbox transmission uncertainty can cause further residuals in desired command torques [**?**]. This error comes from the phenomenon known as gearbox twist, which is the phase difference between motor angle and link angle after considering harmonic drive reduction. In addition to gearbox twist, current measurements are likely noisy, increasing the variance in command torque calculations. The remainder of this section will highlight literature that uses pure dynamics as a basis for solving external wrenches.

An example of using the dynamics directly can be seen in [**?**], where external torque estimation without a force torque sensor by using 12 hall effect sensors (2 per joint) and a current sensor for each motor. The paper specifies the motor current amplitude used based on a sinusoidal waveform with a phase a and phase b instantaneous current

$$I_a = |\frac{i_a}{\sqrt{2}sin(atan(\frac{-3\sqrt{i_a}}{2i_b - i_a}))}|. \tag{7}$$

The torque estimation was done on an ABB IRB 2400-16 industrial manipulator. The parameters in the dynamics equation were found using damped least squares (loss of rank with simple least squares). The formulation of the least squares problem from unknowns in (2) is

$$\tau_c = f(q, \dot{q}, \ddot{q}), \tag{8}$$

where the equivalent solvable expression can be described as

$$\tau_{c,j} = A_j \phi_j, \tag{9}$$

with the damped least squares solution having the form

$$\phi = (A^T A + \lambda^2 I)^{-1} A^T (\tau_c - A\phi_0). \tag{10}$$

The study performed a point-to-point trajectory experiment with an external force acting on the arm and found an average error of 2.47kg (12.35%) of payload when compared to an onboard force sensor (see Figure 3).

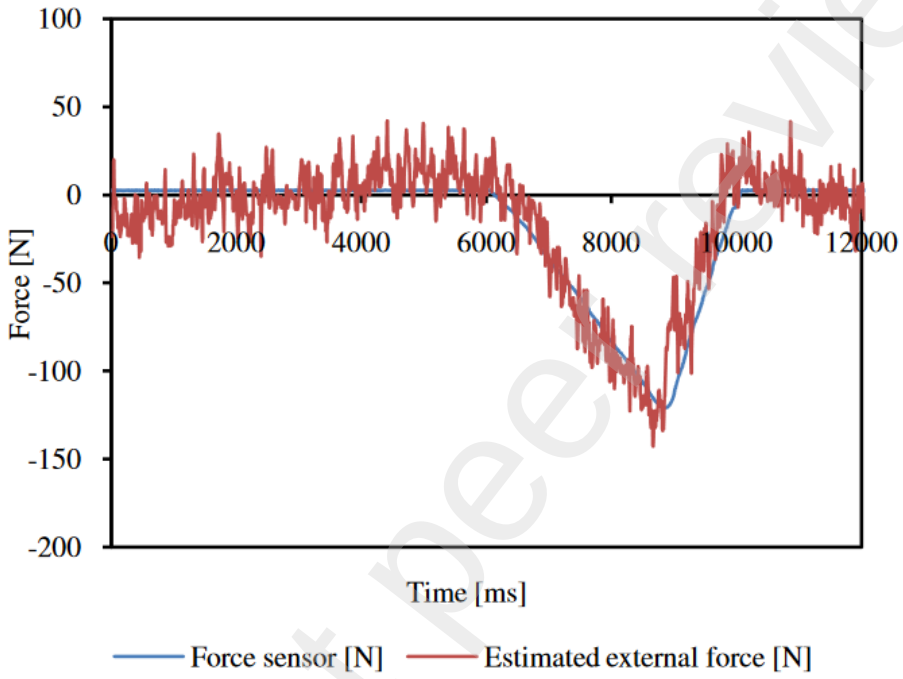

Figure 3: External force estimation using dynamics directly [**?**].

Virtual sensors for estimating the force and motion of a contact point for rehabilitation robots are proposed in [**?**]. Solutions to kinematic and dynamic mathematical models for motion and force, respectively, combined with low-cost position sensors, specifically optical encoders and potentiometers, make up the virtual sensing methods. The proposed methods yielded acceptable results with a mean position error of 0.005 m and a mean contact force error of 1.5 N compared to true measurements.

3.1.2. Human contact model

Virtual sensing for human-robot collaboration safety is studied in [**?**]. The virtual sensor proposed in the paper estimates expected human-robot collision forces and pressures at a given time step using a one-layered nonlinear contact model, with the motivation of complying with the ISO/TS 15066

safety threshold. The model estimates penetration forces on human skin, a highly nonlinear physical phenomenon. The model is described by

$$F_c = \sum_{i=0}^{n} \lambda P \frac{\delta_i}{h} \Delta A_i, \tag{11}$$

where F_c is the collision force, λ is the varying factor depending on indentation shape, P is the contact pressure, δ_i is the infinitesimal deformation, h is the thickness of the skin in contact, and ΔA_i is the contact area. Note that this method only considers the normal force caused by contact. The virtual sensor signals were compared to a well-established finite element model and experimental data showcasing good estimation capabilities of collision forces and peak pressures.

3.2. Momentum Observers For External Contacts

An alternative for wrench estimation that avoids the mentioned drawbacks of requiring $\ddot{q}$ is using the *general momentum* formulation as the basis for external contact estimation. The following sections present various observers that utilize the general momentum equations.

3.2.1. First-Order-Momentum Observer

The first-order-momentum observer (FOMO) was initially proposed in [**?**]. Given the popularity of the general momentum term for external contact estimation, it's worth deriving the FOMO as it works as the basis for methods outlined later. Starting with the generalized momentum of a robot

$$p(q, \dot{q}) = M(q)\dot{q}, \tag{12}$$

where $p \in \mathbb{R}^N$ is the generalized momentum term. Taking the derivative we arrive at (dropping input notation)

$$\dot{p} = M\ddot{q} + \dot{M}\dot{q}. \tag{13}$$

rearranging for $\ddot{q}$ in (2) and subbing into (13)

$$\dot{p} = \tau_c - C\dot{q} - g - \tau_{ext} - \tau_f + \dot{M}\dot{q}. \tag{14}$$

Invoking the *passivity property* to further simplify (14) we get

$$\dot{p} = \tau_c + C^T\dot{q} - g - \tau_{ext} - \tau_f. \tag{15}$$

Using a first-order filter residual vector (r)

$$\dot{r} = -K(r - \tau_{ext}), \tag{16}$$

where K is a gain matrix and for sufficiently large gains we can infer

$$\hat{\tau}_{ext} = r \approx \tau_{ext}. \tag{17}$$

Subbing into (15) and solving for r we arrive at the FOMO formulation

$$r(t) = K\left[p - \int_0^t \left[\tau_c + C^T\dot{q} - g + r - \tau_f\right] dt - p_0\right]. \tag{18}$$

After calibration, the above equations serve as the virtual sensor for external contacts on the end-effector [**?**]. In the frequency domain, the transfer function is defined as

$$\frac{r(s)}{\hat{\tau}_{ext}(s)} = \frac{K}{s + K}. \tag{19}$$

The remainder of the section will outline the application of the FOMO and extensions using the general dynamics form. To save costs on robotic arm manufacturing, [**?**] investigates a sensorless design for sensing forces applied to the end-effector of a robotic arm by using the FOMO. Low-cost hall effect sensors and current sensors were embedded in the joint modules of the robot arm used in the study. The gain for the filter was calibrated by hanging a 5kg load from one of the links in a static configuration and then tuned until the anticipated gravitational forces were accurately estimated. The study follows up with a comparative analysis of the virtual sensor with a real force-torque sensor mounted on the end-effector and found 10% - 15% mean error along each axis. In addition, the observer failed to detect torques less than 10 Nm (likely due to static friction). A correlation of 0.75 between estimated and true contact forces on the end-effector was found.

3.2.2. Band-Pass-Momentum Observer

A virtual sensor for contact or collision detection for industrial robots is proposed in [**?**]. The sensor can distinguish between contact and collision forces by applying low-pass and band-pass filters based on the FOMO. Signals from the low-pass filter represent contact forces by assuming a slow change in force over time. On the contrary, signals that resemble an impulse response closer represent a collision and are output by the high-pass filter. A joint

torque sensor feedback was not used so motor torque is calculated with (6). The low-pass observer is the FOMO directly defined in (18). The band-pass observer is a modified version of the FOMO described as

$$\frac{r_{bp}(s)}{\hat{\tau}_{ext}(s)} = \frac{K_1 s}{s^2 + (K_1 + K_2)s + K_3}, \tag{20}$$

where r_{bp} is the band-pass residual, K_1, K_2, and K_3 are tunable gain parameters. Experiments on a 6-dof robot arm yielded good predictions for collision, contact detection, and differentiation between the two for various tasks.

3.2.3. Second-Order-Momentum Observer

The second-order momentum observer is outlined in [**?**] and is compared with FOMO and a proposed improved second-order momentum observer (ISOMO) for external torque sensing in a 6-dof cobot. The SOMO extends (18) through an additional gain by

$$r_{so}(t) = K_1 \left[p - \int_0^t \left[\tau_c + C^T \dot{q} - g + r_{so} - K_2 r_{so} \right] dt \right]. \tag{21}$$

The formulation for the SOMO in the frequency domain is represented as

$$\frac{r_{so}(s)}{\hat{\tau}_{ext}(s)} = \frac{K_1}{s^2 + K_1 K_2 s + K_1}. \tag{22}$$

SOMO has advantages over FOMO by having anti-interference properties, however, according to the study, a downside of using SOMO is the phase delay compared to FOMO, an important metric in a real-time collision detection scenario. To this end, a compensation link making up the ISOMO is added to assist with the lag in collision detection seen in the SOMO. The compensation link is described as

$$u_f = \int_0^t (\tau_c + h(q, \dot{q}))dt - p, \tag{23}$$

which is directly added to equation (21) scaled by a tunable gain K_u. The ISOMO in the frequency domain is represented as

$$\frac{r_{iso}(s)}{\hat{\tau}_{ext}(s)} = \frac{K_u s + K_1}{s^2 + K_1 K_2 s + K_1}. \tag{24}$$

Overall, the study concludes with ISOMO showing the highest accuracy but stalling in collision detection and anti-interference against FOMO and SOMO respectively (see Figure 4).

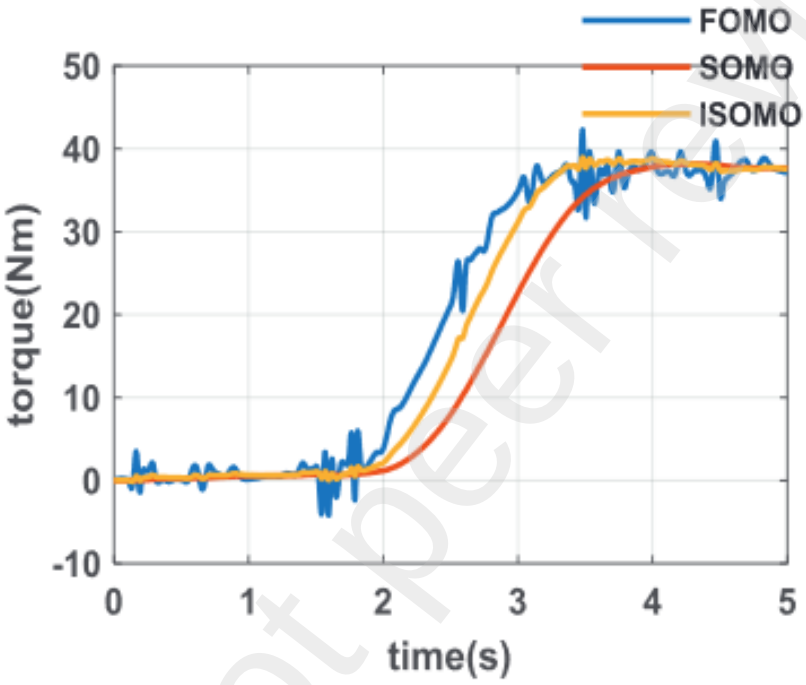

Figure 4: Step response simulation of FOMO (blue); response fastest, and is the noisiest, SOMO (red); has the least noise but struggles with rise time, and ISOMO (yellow); less noise than FOMO, faster than SOMO [**?**].

3.2.4. Sliding-Mode-Momentum Observer

The sliding mode observer (SOSM) and a second-order sliding mode observer (SOSML) are proposed in [**?**]. The observer formulated as

$$\dot{\hat{p}} = \tau + C^T\dot{q} - g - T_1\sqrt{|\tilde{p}|}sgn(\tilde{p}) + \sigma, \tag{25}$$

$$\dot{\sigma} = -S_1 sgn(\tilde{p}). \tag{26}$$

The error between the generalized momentum and SOSM is defined as

$$\dot{\tilde{p}} = -T_1\sqrt{|\tilde{p}|}sgn(\tilde{p}) + s, \tag{27}$$

$$\dot{s} = -S_1 sgn(\tilde{p}) - \dot{\tau}_{ext}, \tag{28}$$

where $\tilde{p} = \hat{p} - p$ and T_1, S_1 are gain matricies. The SOSM differs from the FOMO by directly estimating the generalized momentum and external torques versus using a residual. The SOSML is formulated as

$$\dot{\hat{p}} = \tau + C^T \dot{q} - g - T_1 \sqrt{|\tilde{p}|} sgn(\tilde{p}) + -T_2 \tilde{p} + \sigma, \tag{29}$$

$$\dot{\sigma} = -S_1 sgn(\tilde{p}) - S_2 \tilde{p}. \tag{30}$$

The study did a comparative analysis between the two mentioned observers and FOMO, summarizing that SOSML outperforms the SOSM and FOMO. The SOSM has a drawback in that it converges to the true value linearly with time, while the classical FOMO converges exponentially. However, the second-order variant (SOSML) was able to converge exponentially due to the additional linear terms. In addition, the SOSML had less noise amplification while increasing gain values and yielded slightly better target force tracking than the state-of-the-art FOMO as seen in Figure 5.

3.3. Kalman Filters for External Contacts

Other observers such as various Kalman filters can be used as a virtual sensor for external contacts. Although some extensions are more computationally intensive than the momentum observer (e.g. UKF generating sigma points $O(n^2)$), there is confidence in a near-optimal solution due to the fundamental properties of the Kalman filter as its the optimal state estimator for linear systems, that is to say, there will always be convergence to an optimal solution for a given system in a finite period. Traditionally, the KF mitigates noise from measurements and uncertainty in system dynamics, given variance knowledge of process and measurement noise parameters and can estimate system state parameters without directly measuring the state. However, the statement of optimality only holds if the system is linear but real-world robotic systems are non-linear due to the trigonometric functions found in the dynamics matrices, hence why it's a near-optimal solution because some form of linearization must be done before evaluating the KF leading to inaccuracies in the system model.

3.3.1. Standard Kalman Filter

Although KF extensions can vary, it's worth defining the standard KF to understand the fundamental steps to state estimation. The KF can be

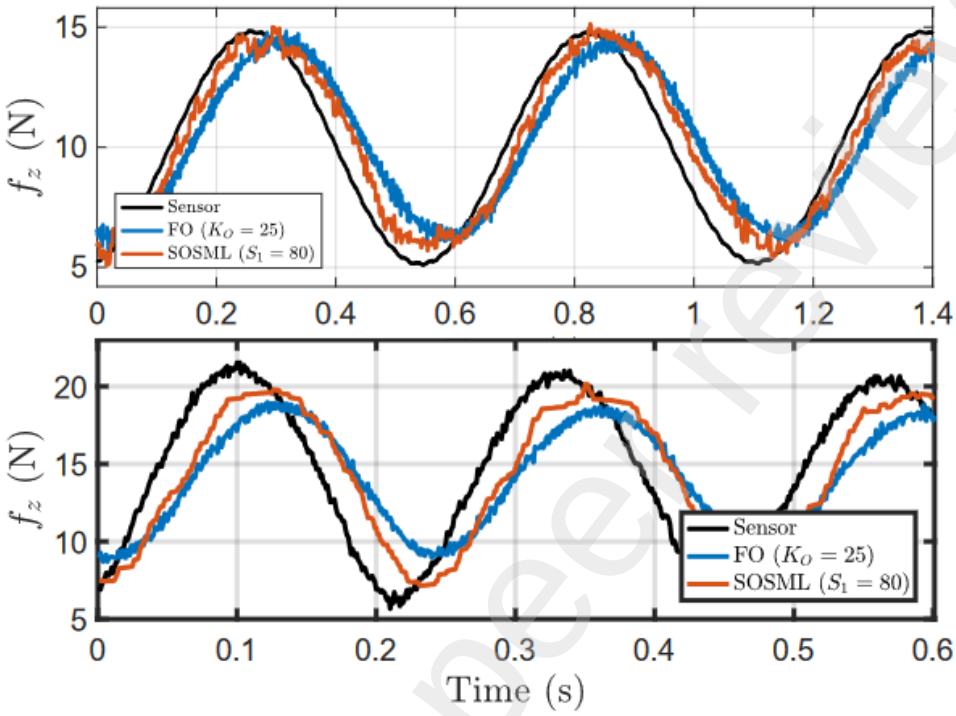

Figure 5: Observed F_{ext} using FOMO (labelled FO), SOSML, and actual force measurements for two mass-spring experiments [**?**].

broken down into three steps, the first step is known as the prediction and is formulated as

$$\hat{x}^+_{k+1} = A\hat{x}_k + Bu_k, \tag{31}$$

$$\hat{P}^+_{k+1} = A\hat{P}_k A^T + Q, \tag{32}$$

where at step k, $\hat{x}^+$ is the state prediction (*a priori*), $\hat{x}$ is the state estimate (*a posteriori*), A is the dynamics matrix, B is the input matrix, u is the input vector, $\hat{P}^+$ is the state-error covariance prediction, $\hat{P}$ is the state-error covariance estimate, and Q is the process noise covariance matrix. The second step is finding the Kalman gain

$$S_{k+1} = CP^+_{k+1}C^T + R, \tag{33}$$

$$K_{k+1} = P_{k+1}^{+} C^T S_{k+1}^{-1}, \tag{34}$$

where S is the innovation covariance, C is the measurement matrix, R is the measurement noise covariance matrix, and K is the Kalman gain. Finally the update or correction step

$$\hat{x}_{k+1} = x_{k+1}^{+} + K_{k+1}(z_k - C x_{k+1}^{+}) \tag{35}$$

$$\hat{P}_{k+1} = (I - K_{k+1} C) P_{k+1}^{+} \tag{36}$$

where z is the measurement vector [**?**]. The remainder of the section outlines the application of various KF formulations for external torque estimation. [**?**] proposes a contact force estimation method using a standard KF. In the study, the momentum of the arm was taken as the estimated state. To estimate the external torque, the state vector was augmented to include the TCP force/moment. Also, the momentum equation (14) was rearranged for command torque and set as known input to the system. The state and input vector are defined as

$$x = \begin{bmatrix} \hat{p} \\ \hat{\mathcal{F}} \end{bmatrix}, \quad u = \begin{bmatrix} \bar{\tau}_1 \end{bmatrix}, \tag{37}$$

$$\bar{\tau}_1 = \tau_c + C^T \dot{q} - g + \tau_f, \tag{38}$$

where the remaining terms are calculated with measured $q, \dot{q}$ and set as input to the system. The following outlines the state space formulation for the system

$$A = \begin{bmatrix} 0_{N\times N} & J^T(q) \\ 0_{N_w \times N} & 0_{N_w \times N_w} \end{bmatrix}, \quad B = \begin{bmatrix} I_{N\times N} \\ 0_{N_w \times N} \end{bmatrix}, \tag{39}$$

where N is the number of degrees of freedom and N_w is the size of the wrench vector (5). The paper differs through dynamic process variance values that depend on the speed of a particular joint. This was done to account for the uncertainty caused by inaccuracies in joint friction modelling, which has more of an effect on the dynamics at low speeds, hence the change in variance. In addition, a comparison is made with the FOMO, and it is concluded that the KF approach produces significantly fewer errors after performing admittance control experiments involving linear point-to-point

movement with an obstruction in between (sustained contact trial). In the experiments, the two approaches were used to determine the contact wrench when the robot made contact with the obstruction and in free motion. Also, experiments were done on a 7-dof manipulator; therefore, to account for the redundancy a Moore-Penrose inverse was found for conversions from torque to force and vice versa.

[?] applies an adaptive Kalman filter (AKF) for force/torque sensing on a UR5 cobot. The AKF works like a classical KF but differs by adding a time-based mode switching average of the input current used for finding the joint command torque (Figure 6).

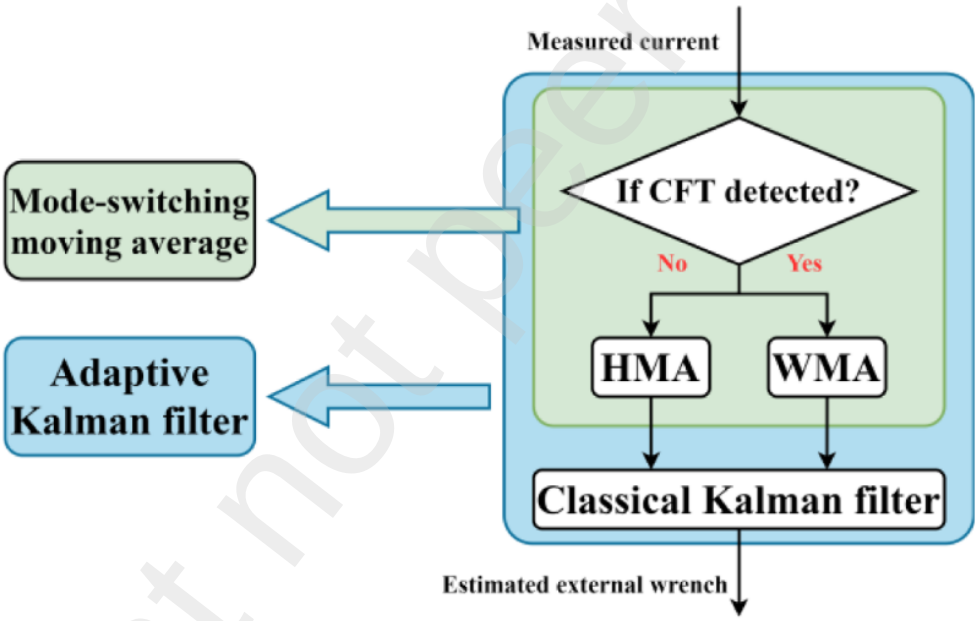

Figure 6: High-level state diagram of the AKF [?].

The formulation of the state is similar to [?], however, with additional adaptive properties. Depending on whether a contact force/torque was detected, the AKF will switch between a Hull moving average (HMA) and a weighted moving average (WMA). Various pick and place experiments were conducted; overall, the AKF performed better than the classical KF for tracking the external wrench when HMA was used (Figure 7). In addition, since

the dynamics uses the generalized momentum, acceleration is not required. This avoids noise associated with numerical differentiation and the computational load associated with computing the inverse of the mass matrix for the acceleration estimate.

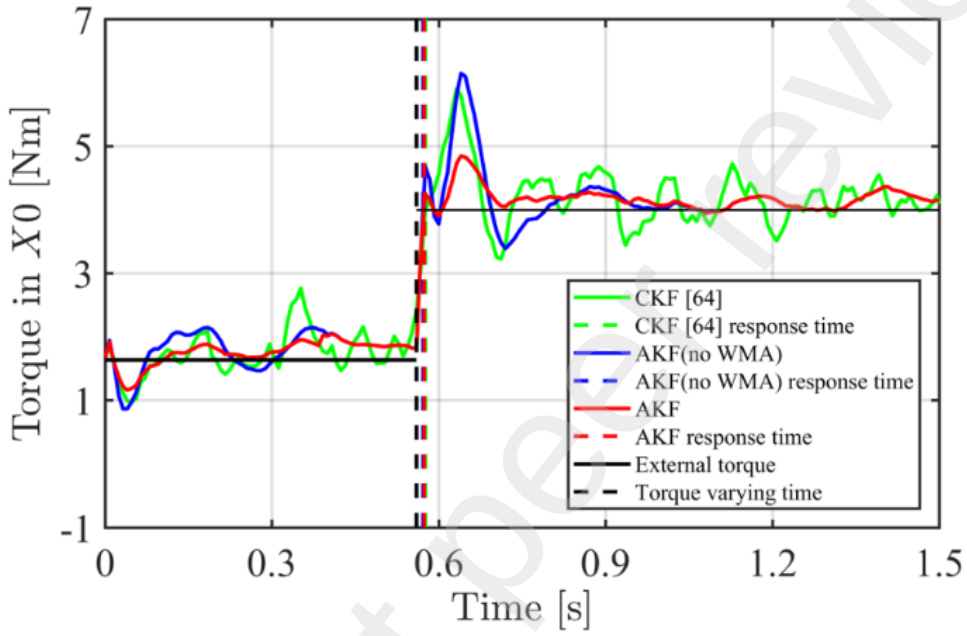

Figure 7: AKF vs. Classical KF for a 1kg load pick up task [**?**].

3.3.2. Extended Kalman Filter

The extended Kalman filter (EKF) is an extension of the standard KF that accounts for non-linear systems by taking a linearization step around an equilibrium point at every time step. This is done by taking the Jacobian of the non-linear state transition ($f(x,u)$) and measurement functions ($h(x)$) for a first-order Taylor series approximation. The following is the EKF algorithm

- *Linearization*:

$$F_{k+1} = \frac{\partial f_n}{\partial x_{n,k}}, \quad H_{k+1} = \frac{\partial h_n}{\partial x_{n,k}}. \tag{40}$$

- *Preditction*:

$$\hat{x}^+_{k+1} = f(x,u), \quad \hat{P}^+_{k+1} = F_{k+1}\hat{P}_k F^T_{k+1} + Q. \tag{41}$$

- *Update*:

$$S_{k+1} = H_{k+1} P_{k+1}^{+} H_{k+1}^{T} + R, \tag{42}$$

$$K_{k+1} = P_{k+1}^{+} H_{k+1}^{T} S_{k+1}^{-1}, \tag{43}$$

$$\hat{x}_{k+1} = x_{k+1}^{+} + K_{k+1}(z_k - H_{k+1} x_{k+1}^{+}), \tag{44}$$

$$\hat{P}_{k+1} = (I - K_{k+1} H_{k+1}) P_{k+1}^{+}, \tag{45}$$

where F is the linearized state-transition function and H is the lineraized measurement function in matrix form to represent the system dynamics and output respectively.

The authors in

[**?**] compare model-based approaches with a model-free approach for external force/torque sensing using only low-cost encoders on a variable radius pulley and stiffness joint mechanisms. The model-based approaches include the EKF and FOMO. The EKF estimates external force by augmenting the state vector with torque and force states. The state vector used by the EKF is defined as

$$x = \left[\hat{q}, \hat{\dot{q}}, \hat{\ddot{q}}, \hat{T}_1, \hat{T}_2, \hat{F}_{tcp}\right]^T, \tag{46}$$

where $\hat{T}_1$ and $\hat{T}_2$, are tension force estimates in the variable radius pulleys. More details on the system state space model can be found in the study. The FOMO combined with an extraction operation, determines additional tension in the experimental mechanism based on a reference input force. Both methods performed similarly, each with their advantages. At times, the EKF was not able to reach peak force measurements, however, the EKF was better at maintaining near zero force estimated when no force (less noise) was applied when compared to FOMO.

3.3.3. Unscented Kalman Filter

The Unscented Kalman filter (UKF) is another extension of the standard KF for non-linear state estimation. The UKF is unique in that, instead of linearizing the state-transition function directly (like the EKF) the distribution generated by the state-transition function is approximated by a Gaussian distribution, a necessary condition for optimal state-estimation using Kalman filtering techniques. This is done by creating a set of known points called *sigma points* that are then propagated through the state transition function. The propagated sigma points are then used to reconstruct a Gaussian

approximation of the non-linear distribution formed by the state-transition function. This operation is called the *unscented transform*.

- *Generate sigma points & weights*:

$$\chi_k^0 = \tilde{x_k}, \tag{47}$$

$$\chi_k^i = \tilde{x_k} + (\sqrt{(N+\lambda)P_k})_i, \tag{48}$$

$$\chi_k^{i+N} = \tilde{x_k} - (\sqrt{(N+\lambda)P_k})_i. \tag{49}$$

  Constant λ is a tunable scaling parameter and χ are the sigma points. The weights W_i of the sigma points used to calculate the mean of the state can be defined as

$$W_0 = \frac{\lambda}{n+\lambda}, \tag{50}$$

$$W_i = \frac{1}{2(n+\lambda)}. \tag{51}$$

- *Prediction*: Using propagating sigma points through $f(x)$, one can find the *priori* state estimate and state error covariance as follows

$$\chi_{k+1}^i = f(\chi_k^i), \tag{52}$$

$$\tilde{x}_{k+1}^+ = \sum_{i=0}^{2N} W_i \chi_{k+1}^i, \tag{53}$$

$$P_{k+1}^+ = Q_k + \sum_{i=0}^{2N} W_i [\chi_{k+1}^i - \tilde{x}_{k+1}^+][\chi_{k+1}^i - \tilde{x}_{k+1}^+]^T. \tag{54}$$

  The measurement predictions are based on the propagating sigma points through the non-linear output function $h(x)$

$$\gamma_{k+1}^i = h(\chi_k^i), \tag{55}$$

$$\tilde{y}_{k+1} = \sum_{i=0}^{2N} W_i \gamma_{k+1}^i. \tag{56}$$

  where γ are the measurement sigma points.

- *Update*

$$P_{yy,k+1} = \sum_{i=0}^{2N} W_i[\gamma_{k+1}^i - \tilde{y}_{k+1}]][\gamma_{k+1}^i - \tilde{y}_{k+1}]^T, \tag{57}$$

$$P_{xy,k+1} = \sum_{i=0}^{2N} W_i[\chi_{k+1}^i - \tilde{x}_{k+1}^+][\gamma_{k+1}^i - \tilde{y}_{k+1}]^T, \tag{58}$$

$$K_{k+1} = P_{xy,k+1} P_{yy,k+1}^{-1}, \tag{59}$$

$$\tilde{x}_{k+1} = \tilde{x}_{k+1}^+ + K_{k+1}(y_{k+1} - \tilde{y}_{k+1}), \tag{60}$$

$$P_{k+1} = P_{k+1}^+ - K_{k+1} P_{yy,k+1} K_{k+1}^T. \tag{61}$$

The update stage requires the corrective gain K_{k+1} which can be calculated using the innovation covariance $P_{yy,k+1}$ and the cross-covariance $P_{xy,k+1}$, followed by updating the state vector and state error covariance [**?**].

The authors in [**?**] applied a UKF to a humanoid robot for estimating external torque among other states. The goal of the study was to enable torque control on the robot without relying on joint torque sensors. Instead, it took a sensor fusion approach utilizing all onboard sensors in the UKF measurement model. The robot included encoders, force/torque, inertial (accelerometer, gyroscope), and motor current sensors. The state and measurement vector are

$$x = \left[\hat{\dot{q}}, \hat{\tau}_c, \hat{\tau}_f, \hat{f}_{FT}, \hat{F}_{ext}\right]^T, \tag{62}$$

$$y = \left[\dot{q}, I_c, f_{FT}, \alpha_{acc}, \omega_{gyro}\right], \tag{63}$$

respectively. Note that this is a simplified version only considering one instance of a joint for each state and measurement. The actual state and measurement vector used is much larger to account for multiple joints and sensors on the robot, more details on the process and measurement model dynamics along with the state transition function used can be seen in the study. The UKF was compared to the RNEA method for finding joint torques and performed significantly better at tracking random disturbances. Although not a robotic arm, the principles outlined can certainly be applied to a robot arm, given the complexity of a humanoid robot.

3.4. Machine Learning & Deep Learning Approach

Machine learning (ML) and deep learning (DL) methods for external contact virtual sensing is prevalent in the literature.

The authors in [**?**] reformulated the problem of sensing one contact force by proposing a no-zero probability that two contact points can create the same torque given a set of circumstances. Therefore, a possible solution set is found based on joint-torque sensor measurements. The study then combines rejection sampling for measurements below a threshold and gradient descent to find a potential optimal solution to the non-linear optimization problem for finding the true contact point. When a contact force is detected the robot takes an exploration step to determine whether the detected force is a collision or contact. If it's a contact then a movement in the opposite direction should relieve the force, if it's a collision then the force will remain unaffected (in most cases).

The results in [**?**] take a machine-learning approach for external torque estimation on a UR5 robot arm. The study takes advantage of all the onboard sensors (105 in total) and takes all of the readings as *features*. With this data, transfer entropy (TE) is applied to reduce the dimensionality of the feature space. The purpose of using TE is to determine which robot sensors are most influential when a preconfigured external torque is applied. Two reduced feature spaces were found using this method. The first focuses on a phase shift caused by external torques, and the second for predicting the external torque value. Once found, a further dimensionality reduction was done using principal component analysis. A model was then trained on the two feature spaces for the respective estimation. In a real-time setting, all recorded data is projected into these feature spaces, estimating the motor phase and external torque. The approach was validated through nut tightening experiments to precise specification and concluded quality torque estimates, however, the accuracy was time and phase-dependent.

The results found in [**?**] take a probabilistic approach to contact point estimation and a proposed control scheme. In addition, robot and environment geometry must be estimated in parallel, given a set of sampled joint positions. The study derives a least squares formulation from an initial equation using the Bayes rule

$$Bell = p(s|q^t, r) = p(q^t|r, s)\frac{p(s|r)}{p(q^t|r)}, \tag{64}$$

where $q^t = [q_1, q_2, \ldots, q_m]$ is the set of sampled joint angles, s is the set of parameters encoding the environment, and r is the set of parameters encoding the robot shape. The same formulation can be put into a least squares formulation represented as

$$log(Bell) = -\frac{1}{2\sigma^2}\sum_{i=1}^{t} dist(r, s, q_i)^2 + c \tag{65}$$

where $dist(r, s, q_i)$ represents a Gaussian distribution of the Euclidean distance between the robot and object of contact and c is a constant that results from taking the log of the $Bell$ equation. A note on computational time; the parameters representing the robot's shape (r) only need to be calculated once. The parameters were found using a non-linear optimization technique. The initial parameters for the optimization were found using physical measurements with the motivation of bypass local minima, improving chances of convergence to true values. Experiments were conducted to simulate a single point of contact on the edge of a link, effectively containing the motion to a plane and representing the environment s by a single contact point. Overall, single contact points were effectively estimated, dramatically benefiting the step force response of the system as shown in the study. The study notes that sufficient degrees of freedom are required to accurately predict contact points, as exploratory motions around the contact point are necessary when taking the probabilistic approach mentioned.

Revisiting [**?**], a model-free approach was also taken for external contact estimation. The study uses an ANN, specifically an Long Short-Term Memory (LSTM) model, to predict external forces. After several tests, the ANN yielded the best estimation performance regarding RMSE compared to the previous two model-based (FOMO and EKF) approaches.

3.5. Fusion of Methods

This section will outline literature that combines approaches for virtual external contact sensing. Several methods for external force and moments using virtual sensing on a 7-dof robot arm are proposed and tested in [**?**]. One method involves sensing external torque without knowledge of the contact point, but using measurements from a force-torque sensor located at the base. The external force is found using

$$\mathcal{F} = \mathcal{F}_m - \mathcal{F}_g, \tag{66}$$

where $\mathcal{F}_m$ represents measurements from the force-torque sensor while $\mathcal{F}_g$ represents the forces on the robot when no external contacts are being applied. This value is calculated using the recursive Newton-Euler algorithm (RNEA). However, the RNEA requires joint accelerations which are found using the FOMO (estimating $\ddot{q}$ by subbing $\hat{\tau}_{ext}$ into (2)). Further scenarios are tested, specifically, with a force sensor at the base, and no sensor at all. Having measurements from the force sensor improved accuracy of estimation, however, both scenarios yielded promising estimates for force values and contact point location. The study later details the extraction of external moments if a contact point estimate is provided.

[**?**] proposes a virtual sensor for contact force estimation. The sensors can identify the contact force using the FOMO paired with (Equation 4) and can track the contact point by using a depth sensor and image processing algorithms. The method can track up to two contact points at a time (Figure 8). Static experiments were conducted with a KUKA LWR to validate the virtual sensor's performance yielding good performance when compared to true force measurements ($RMSE_{xyz} = [0.95\%, 5.03\%, 2.02\%]$).

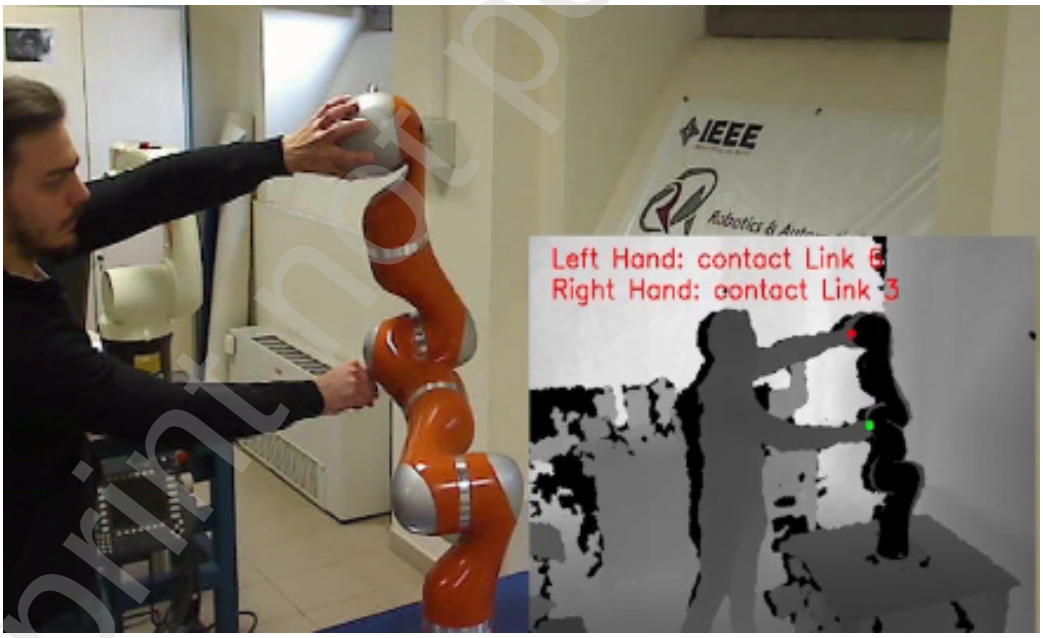

Figure 8: Contact point estimation with computer vision on KUKA LWR [**?**].

[**?**] uses the FOMO for external torque estimation. However, the study

proposed a fusion function for two observers which were given separate joint-torque measurements, one from motor current and the other from torsional deformation of the joint. This deformation is the difference between the motor and the link position. Pairing this with a linear spring and viscous friction model, an estimated joint torque is found. Two approaches are taken for fusion function formulation. The first approach finds constant weights derived from Bayes' theorem by fusing both measurement distributions, leading to a smaller overall variance. The second approach uses adaptive weights where the variances are replaced with the square of external torque measurements, the latter approach yielded lower RMSE values. Validation sets showed that the compensation of torsional deformation yielded much lower root mean squared error (RMSE) values when compared to its absence. The fusion of the deformation case with the motor current torque estimates proved slightly better than strictly sticking to the motor current torque estimate.

[**?**] utilizes LSTM neural networks to fine-tune the external torque estimated with the FOMO. An LSTM was the chosen model due to its time-series properties, which better represented the hysteretic properties of friction. The LSTM uses a many-to-one structure with 1 hidden layer composed of 20 hidden units for each state. The data buffer sequence of the LSTM was chosen to be 100 units. The study defines an uncertainty torque as

$$\tau_u = \tau_f - \tau_e, \tag{67}$$

where τ_e is modeling error (inaccurate inertia tensors, center-of-mass (COM), etc.). The uncertainty torque is added to the general dynamics and is used in the FOMO computation. The LSTM is trained to find τ_u when the 2-dof test platform is executing free motion (i.e. no τ_{ext} present), under these conditions, the FOMO will output a low-pass filtered version of the uncertainty. During a real collision/contact case, the external torque is found by subtracting the trained LSTM model prediction from the FOMO estimate given the current joint angular position and velocity measurements. The method was tested against various modelling errors, including a mass error (reducing the mass of each link), a COM error, and a combination of the two. The study found that the LSTM failed to capture the effects of static friction and poor performance for uncertainty predictions when the robot rarely moved (friction was assumed in the sliding region). This suggests the potential to apply a hybrid approach to the problem, merging well-defined static friction models with the LSTM model to better account for uncertainties in the

model. However, under normal operation, the proposed method was able to sense external torques with the FOMO and account for friction and modeling errors scenarios with the trained LSTM model.

[**?**] proposes a disturbance KF (DKF) which takes a similar formulation to [**?**], utilizing the general momentum equation with a few modifications, one of which is an additional nonparametric term for error compensation caused by inaccuracies in dynamics parameters used in the inertia, Coriolis, and gravity terms. Specifically, non-linear effects produced by motor backlash and internal joint friction were compensated for by training a simple multilayer perceptron (MLP) model with joint position and velocities as input and modelling error as the target. Building off the generalized momentum derivation with the additional trained MLP compensation term $N(q, \dot{q}) \in \mathbb{R}^N$

$$\dot{p} = \tau_c + C^T(q, \dot{q})\dot{q} - g(q) - \tau_{ext} + N(q, \dot{q}). \tag{68}$$

The state and input vector are defined as

$$x = \begin{bmatrix} \hat{p} \\ \hat{\omega} \end{bmatrix}, \quad u = \begin{bmatrix} \bar{\tau}_2 \end{bmatrix}, \tag{69}$$

$$\bar{\tau}_2 = \tau_c + C^T\dot{q} - g + N(q, \dot{q}), \tag{70}$$

where $\hat{\omega} \in \mathbb{R}^N$ represents the dynamics of the disturbance. Assuming a constant or instantaneous disturbance $\hat{\omega} = \hat{\tau}_{ext}$ is true, and the state space is defined as

$$A = \begin{bmatrix} 0_{N\times N} & I_{N\times N} \\ 0_{N\times N} & 0_{N\times N} \end{bmatrix}, \quad B = \begin{bmatrix} I_{N\times N} \\ 0_{N\times N} \end{bmatrix}. \tag{71}$$

The authors tested the DKF against the non-linear disturbance observer (NDOB) and the FOMO. The RMSE results indicated that DKF outperforms the other two methods marginally, and was joint dependent. Figure 9 highlights the variations between the three methods and further supports the RMSE results by showcasing the noise-filtering properties of the DKF.

A recent study by [**?**] tackles the problem of external contact force sensing for reduced costs. The paper details a virtual sensing architecture that utilizes machine learning, Kalman filtering, and physical models to generate signals. The popular ensemble learning algorithm, XGBoost, derives unknown robot parameters for the inverse kinematics of the 6-dof robotic

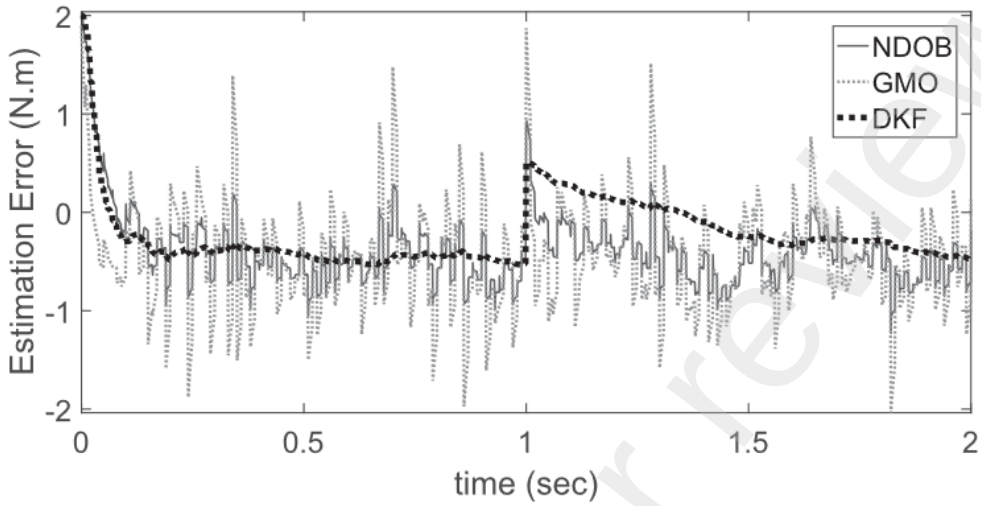

Figure 9: The DKF, NDOB, and FOMO (GMO) disturbance error plot on the second joint [**?**].

arm used in the study. The virtual sensing algorithm uses the predicted inverse kinematics derived from XGBoost to estimate the torque seen on each axis. The following method paired with (4) was used to find contact forces on the end-effector

$$\tau_{ext} = \tau_{act} - \tau_{pred}, \tag{72}$$

where τ_{act} is the actual torque produced by the motor drivers and τ_{pred} is the predicted torque the dynamic model finds under no load. Correlations between the observed torque and measurements using a F/T sensor ranged from 0.71-0.75. Forces less than 20N were not observable which was attributed to static friction.

4. Friction & Uncertainty Torque Virtual Sensors

In addition to the external contact experienced by robotic manipulators, other torques arise that cannot be attributed to a contact. These torques include the friction and uncertainty terms that were briefly mentioned in the previous sections.

4.1. Model-based

Interaction forces between industrial robots and the environment are identified using a virtual sensor in [**?**]. A similar approach to finding external torques to (4) and (72) is used, except the predicted internal joint torques (τ_{pred}) are found using a combination of inertia torques and frictional torques. The inertial torques are represented by

$$\tau_i = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q), \tag{73}$$

The frictional torques are represented with a combination of a third-order polynomial friction model and a Stribeck term. Per joint, the estimated friction is

$$\tau_f = [k_0 + k_1|\dot{q}| + k_2|\dot{q}|^2 + k_3|\dot{q}|^3]sgn(\dot{q}) + k_4 e^{-c_s|\dot{q}|}sgn(\dot{q}), \tag{74}$$

where k_i are polynomial coefficients and c_s is the Stribeck coefficient. Note the parameters are not constant across joints. Non-linear estimators are used to determine the model coefficients stated above. Optimal parameters for an exciting trajectory are found with a genetic algorithm. However, two objective functions are minimized, first for low-velocity where gravitation and static friction are dominant and second for high-velocity where viscous and inertial effects dominate. In addition, the paper takes into account the internal joint temperature through adding a polynomial friction torque that is linearly dependent on temperature. Finding the thermal coefficients of this equation proved computationally intensive, taking several hours to find. The virtual sensor is later verified using an experimental setup and compared to contact force measurements by load cells attached to the gripper mechanism. For local and global trajectories, the virtual sensor had a 4N and 14N accuracy, respectively.

[**?**] proposes a model-based control enhancement by deriving a harmonic drive transmission model. In the proposed model, non-linearities arising from harmonic drives are accounted for through coulomb friction, viscous friction and wave generator torque terms. These terms in addition to a gravity torque are what make up the command torque. This model can enhance dynamic models that solve for τ_{ext} by including friction terms in the general dynamics form.

4.2. Machine Learning & Deep Learning Approach

The authors in [**?**] focus on drift compensation caused by extreme temperatures, specifically, those experienced in space. This paper is important as it highlights the substantial drift associated with F/T sensors when subject to non-ideal conditions. The study aims to be applied to the Chinese Space Station Manipulator System (CSSRMS). There are two main components to the CSSRMS, the core module manipulator and the experimental module manipulator, the latter houses the custom F/T sensor. Experiments ranged from -30 to 60°C to gain an intuition on the relationship between the induced drift and current temperature. The most linear relationship can be seen in Figure 10.

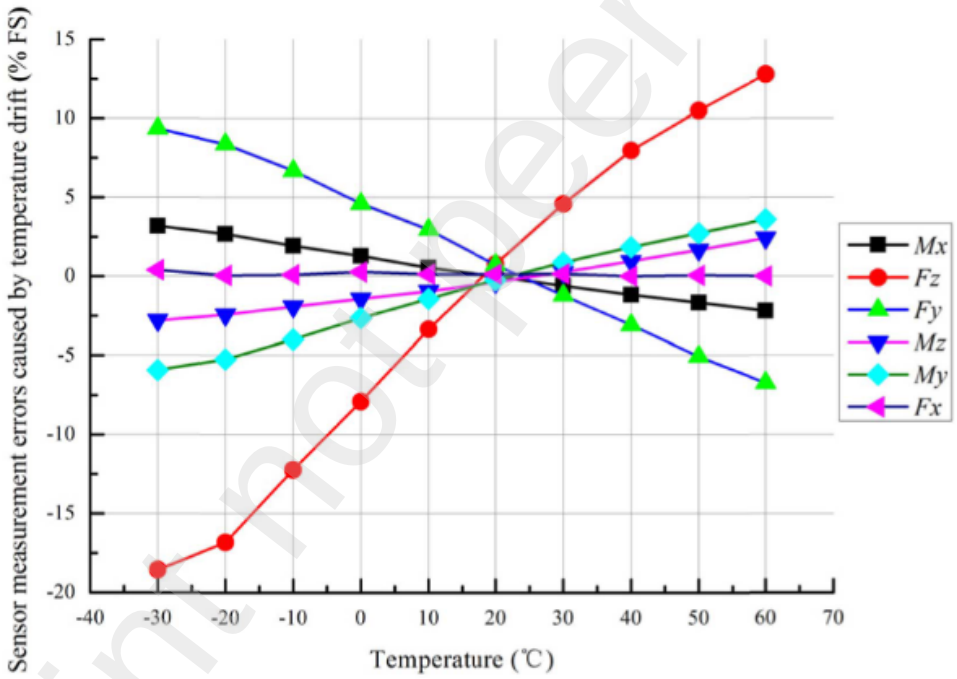

Figure 10: % Full-scale measurement error vs. Temperature [**?**].

At ambient temperature, the F/T sensor was modelled as

$$F = (C^T C)^{-1} C^T V, \tag{75}$$

where C is a vector of strain gauge constants and V is the output voltage of the Wheatstone bridge circuit (which varies when the load is applied). At a high level, the compensator was represented as the inverse of the F/T model but instead varied with temperature and voltage

$$\bar{F} = f^{-1}(V, T). \tag{76}$$

Three methods were used to compensate for the F/T drift. Given the obvious linear relationship, the least squares (LS) approach was used and reduced the error from 20% to just over 2%. The second method used was a radial basis function (RBF) ANN and beat the LS approach by yielding an error of 0.6%. The final approach was a least squares support vector machine (LSSVM) yielding outstanding results of 0.011% error, virtually removing the drift (Figure 11). Hyperparameters of the model were found using the particle swarm optimizer (PSO). More details on the LSSVM formulation can be seen in the study.

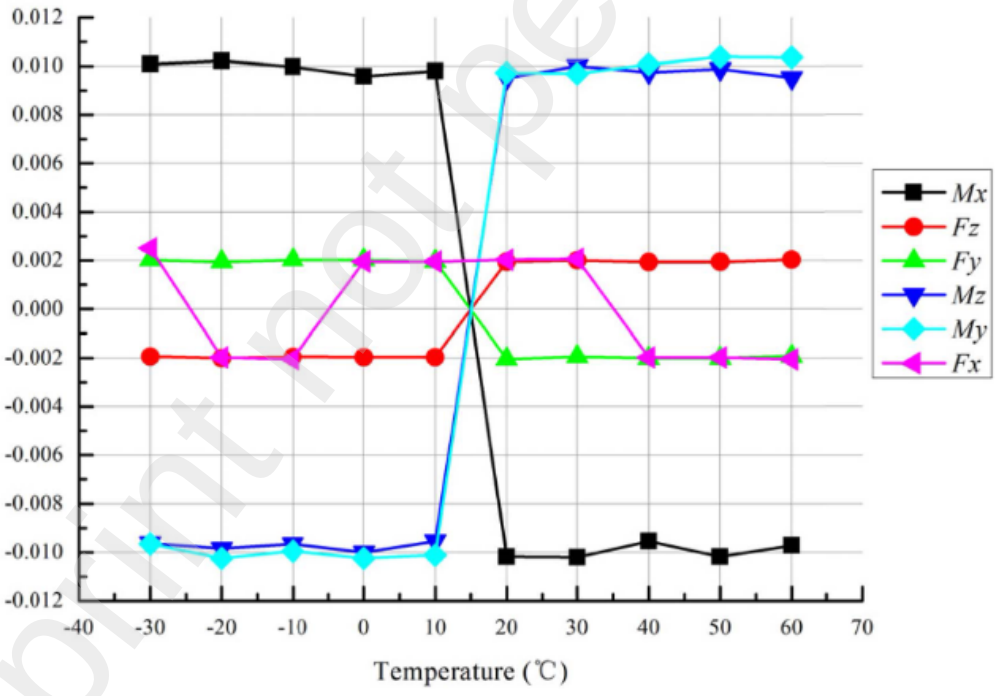

Figure 11: PSO-LSSVM sensor measurements error [**?**].

5. Discussion

5.1. Trends of virtual sensors in robotic arms

5.1.1. Applications

A common application of virtual sensors to robotic arms is the determination of contact forces and external torques at the end-effector or a specific point along the kinematic chain. There are a few unique ways that this is accomplished. A common approximation for external torque is using a model-based approach with the residual of a moment-observer defined in Equation 18 along with its many extensions, in addition to the use of KF techniques in an augmented state vector format. The moment-observer is a popular method applied due to the lack of an acceleration term required when estimating external torques, contrary to the general dynamics equation. This avoids the need for additional sensors to measure the angular acceleration of each joint or double differentiating encoder outputs, which would amplify noise.

From this, two motivations for virtual sensing in robotic arms are derived from the literature: (1) increased safety by implementing virtual sensors for collision detection and (2) reduced total costs by replacing expensive sensors (e.g. force-torque) with their virtual counterpart. Although both motivations are valid and important for the robotics sector, they're mutually exclusive. This is because virtual sensors can only estimate physical measurements, leading to an additional uncertainty to consider when designing a system. If safety is of utmost importance, such as in a space application like structural assembly [**?**] or EVA support, using virtual sensors as redundancies to their physical counterpart is the ideal option as this would ensure a backup in case of a hardware fault or an additional validation layer when performing data processing [**?**]. This is assuming that the physical sensor in question has been flight-proven and the test readiness level matches that of the mission requirement. Alternatively, a threshold achieved accuracy by the virtual sensor can be deemed safe under authority approval and can completely replace the physical counterpart. A convergence of the two motivations could be using lower-cost sensors while achieving the same accuracy by pairing them with a virtual counterpart (potentially a sensor fusion avenue).

5.1.2. Gap in Virtual Sensing

There is a lack of black-box approaches for virtual sensor development. This could be in part due to the difficulty of proving stability with parameters

derived using black-box approaches, which directly impacts the feasibility of a robot in safety-critical environments [**?**]. Applying the appropriate machine learning methods like Physics Informed Neural Networks (PINN) to the virtual sensing architecture can help overcome the ambiguity of black-box approaches as the models are bounded by the physics of the manipulator and environment, potentially mitigating outlier cases [**?**]. The most common use of machine learning and black-box methods in virtual sensor development found in the literature was determining model coefficients, such as in [**?**], or accounting for random uncertainty in dynamics modelling with an additional error term found using a neural network, like in [**?**].

Another reason for the lack of black-box approaches is that some applications of virtual sensing were for measurable parameters that already have a solid mathematical basis, such as frictional torques. However, in less ideal environments where large temperature fluctuations are common, these parameters become unpredictable. Novel approaches to reflect complex and highly non-linear relationships not directly measurable by a physical sensor will inevitably lead to the application of black-box methods. An example of this is highlighted in [**?**] where various machine learning methods are used to account for drift in a 6-axis force/torque sensor when experiencing large temperature fluctuations.

5.2. Challenges in Virtual Sensor Development

Application-specific transferable virtual sensors may become common. For example, two identical robots can use different virtual sensors to measure quantities relevant to the field of application. However, the transferability poses a challenge in virtual sensor development as model parameters vary between robot to robot due to potential manufacturing inconsistencies. To this end, transferability becomes a challenge, especially when deep-learning approaches are used for development due to over-fitting when modelling highly nonlinear relationships in novel environments. A potential solution to the transferability problem is applying larger capacity models that capture a broad range of situations encountered by a robotic manipulator. The said model can then be updated for fine-tuning to the desired robot that is running the virtual sensor algorithm. Larger models has trade offs when computational resources are limited, leading to the next challenge of applying virtual sensors onto real-time systems.

Given the potential complexity for virtual sensing algorithms, using a combination of estimation and machine learning methods, there is the open

question of how to apply these algorithms to real-time controllers due to implementation issues caused by size and computational constraints faced on embedded devices. This is especially true for applications where quick choices are important, such as surgical robots or factory line cobots, where workers are at risk. Methods for compressing large machine learning models, such as pruning and optimization, can help achieve smaller models that fit on embedded devices. To further minimize resource usage, model compression can be paired with a fundamental change to the machine learning algorithm architecture, reducing the number of parameters. For example, replacing computationally intensive support vector machines and gradient boosting machines with decision trees or shallow neural nets [**?**].

5.3. Future of virtual sensors in robotic arms

Novel application of machine learning methods motivates their use in virtual sensor development. An example can be seen in [**?**] where reinforcement learning is used to train a digital twin of an industrial robot arm; a similar approach may be considered for making a virtual sensor. The authors in [**?**] outline a virtual sensor network for fault detection in microcontrollers, which are inevitably responsible for a robot's core functionality. In [**?**], a virtual sensor for object recognition on a mobile robot was developed and applied. As virtual sensors become more common, most robotic arms will utilize some form of nested virtual sensor. For example, a multi-model machine learning architecture can be used to map complex relationships between immeasurable parameters. Once virtual sensors have been validated using a pre-defined metric (accuracy or precision rating), the costs of high-end industrial robotics will be impacted by replacing expensive high-precision force torque sensors with their virtual counterpart. To this end, a virtual sensor can be packaged as a piece of software intellectual property and will serve as a market opportunity for robotics researchers and software developers.

6. Conclusion

Virtual sensors for robot arms and systems yield many benefits over their physical counterparts. Some of these include free redundancy, mirror physical sensors without additional cost, reliability, no wear or drift with time, adaptability, can be added and modified without physical installation or workarounds, and reduced costs. Expensive sensors can be replaced with

accurate and robust virtual sensors. These advantages are especially intriguing in the world of robotic manipulators, as space and mass constraints pose a challenge for the installation of additional sensors along with the costs associated with high-resolution sensors such as force-torque sensors commonly found on robotic end-effectors. The literature focuses on model-based approaches to virtual external contact wrench sensors. This presents the opportunity for novel virtual sensors utilizing black-box approaches along with unique application-specific virtual sensors.