Motion compensation simulated on DaVinci surgical robot

Anand Parwal and Ankur Agrawal

Abstract— **Motion compensation is the process of simulating motion of objects and bringing a robot in synchronization with this motion. This is very helpful in surgery where small deviations can cause lots of damage to surrounding tissues. In this project we explore the implementation of motion compensation algorithms with the help of a low level controller on the Da Vinci surgical robot. We propose to predict the involuntary motion of internal organs like beating of heart using predictive filters such as Kalman filters to compensate for the motion of the organ such that the organ is relatively stationary. This makes the surgical procedure easier and safer as it is performed on a stationary organ. The task is performed in a simulation environment. Simulation results also show how well these motion compensation algorithms work when compared to the actual organ motion.**

I. Introduction

While performing surgery, some organs tend to have involuntary reactions in order to keep the body alive and well. Some of these organs are heart, lungs, and eyes. Along with organ movement, the tissues inside them, own or foreign move as well [1]. During surgery, surgeons compensate this motion by moving the instruments manually along the axis of motion. This gets more complicated when the surgery is performed with a tele-operated robotic arm as the surgeon has to replicate the motion of the organ while performing the procedure. However, if the robot can replicate the motion of the organ, rendering it at rest with respect to the tip, the surgery becomes easier for the surgeon. If the motion compensation is perfect, the precision of the procedure increases in addition to the decreased risk of something going wrong. However, there are always some delays to the motion compensation algorithms and therefore introduce a phase shift between the motion of the organ and that of the robot tip. These phase shifts are undesirable as they can sometimes make the tip move opposite to the organ, causing even further damage to the organ. Moreover, after the compensation, the robot still has to be tele-operated by the surgeon, but now relative to a stationary organ. Hence, the resulting motion of the robotic tip will be a superposition of two motions: the motion compensating the organ motion and the movement done by the surgeon to perform the surgery.

Motion compensation has 2 main steps, prediction and control [4]. Algorithms have been studied mostly for open heart surgeries. In these algorithms, the motion of the organ is seen and the next state is predicted and the robot tip is moved accordingly. A point on the organ is marked with a fiducial and that fiducial is tracked using cameras (endoscopes) or 3D ultrasound [2]. For the case of heart, electrocardiogram (ECG) signals can also be used to track the motion of the heart [4]. After, the state of the organ at a certain time is obtained, the next state is predicted by filters such as Extended Kalman Filters [2], [3] or generalized adaptive filters [5]. Another approach to remove the relative motion between the robot and the organ is by force feedback control as is done in [6]. The force with which the tip presses the organ is controlled. When the organ moves towards the tip, the forces increases and this feedback makes the tip move up. The advantages of this method is that only a certain amount of regulated force is applied on the organ which can be set below the piercing limit of that organ and hence maintaining safety during the procedure. Cameron N. Riviere et al. Used a weighted-frequency Fourier linear combiner (WFLC) to predict the motion and used repetitive control for the end effector [7]. Barthelemy Cagneau et. Al used iterative learning control (ILC) to control the robot after compensation algorithms [8]. Many visual techniques have been used as well ([9]-[11]).

In this project, we propose to study the motion compensation algorithms and apply them in simulation on the DaVinci surgical robot by Intuitive Surgicals incorporation. The platform used for simulating the robot and its motion will be Gazebo using ROS for communication. The DaVinci robot is a 6 DoF arm at the patients side cart and has a 7 DoF control for ergonomics at the surgeons console. The Gazebo models are available open source. The motion is predicted by a kalman filter and the motion of the organ is done using a Fourier series transform which gives a randomized sign wave making it random and hence the system will be robust to compensate for all motions.

II. Preliminaries

A. Kalman Filter

Kalman Filter is a Bayesian predictive filter. It implements belief computation for continuous states only. It represents the belief at any time t using a Gaussian distribution function having a mean μ_t and covariance Σ_t. This mean represents the state with an added uncertainty given by the covariance matrix . A more detailed explanation of Kalman filters can be found at [13].

The Kalman filter algorithm is a recursive algorithm which calculates and updates the mean and the covariance taking into consideration the measurements of the state. The inputs are the previous belief or mean at time $t-1$ and the corresponding covariance matrix, in addition to the measurement of the state z_t at time t. The algorithm calculates something known as Kalman Gain, K_t and uses it to update the state belief. This update takes place in two steps: state transitions, which uses the dynamics of the system to update the belief, and measurement update, which updates the updated state to incorporate current measurements. The algorithm for Kalman filter is summarized below in 1. The matrix A_t represents dynamics of the system and B_t the impact of control u_t has on the state of the system. R_t represents the uncertainty during the state transition, whereas Q_t represents the errors in measurement data.

Algorithm 1 Kalman filter algorithm

procedure KALMAN FILTER(μ_{t-1},Σ_{t-1}, z_t)

$\bar{\mu_t} = A_t \mu_{t-1} + B_t u_t$

$\bar{\Sigma_t} = A_t \Sigma_{t-1} A_t^T + R_t$

$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$

$\mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t)$

$\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t$

return μ_t, Σ_t

Although Kalman filter is quite an effective predictive filter, it has some limitations. It is designed to work for linear systems, where the next state is a linear function of the previous states. Furthermore, the measurement should also be a linear function of the state with added uncertainty. Barring these Markov assumptions, the predictive algorithm loses its efficiency and cannot be proven to be a Gaussian distribution.

III. Methodology

The approach taken in this paper to demonstrate motion compensation on the daVinci surgical is shown in Fig. 1. From this flowchart, it can be seen that there are five main blocks of the project. One of the blocks is to simulate the organ motion. For this project, we are assuming an organ motion with only one degree of freedom. This can be observed with movements of the organs such as lungs and hearts which pre-dominantly move in the anterior direction. Furthermore, considering only one degree of freedom reduces the complexity of the system, so that the algorithm can be developed and later be scaled for higher a degree of freedom system. Here, we consider the organ motion to be a random Fourier series in only the vertical plane of the simulation environment in Gazebo. The purpose of selecting this random function was to not develop the algorithm only for a particular organ motion, but a robust system where in the motion of any organ can be just plugged in.

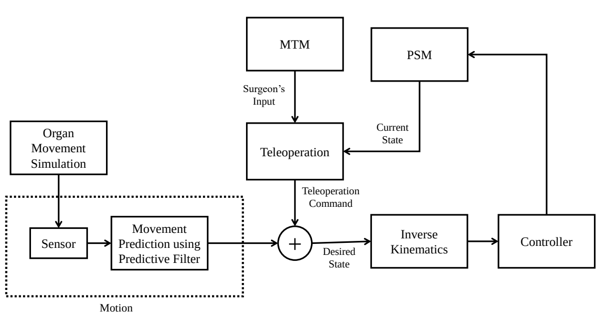

Fig. 1. Flowchart showing the methodology implemented in this project

A sensory input which detects the organ motion is required as an input for our motion compensation algorithm. This input can be either from a camera or 3D ultrasound, or maybe even an electrocardiogram [4]. A predictive filter is then used on this input to predict the next state of the organ. This next state is given to the inverse kinematics of the slave robot which operates on the organ. The master console is not involved in this process.

The role of the master comes when it is being controlled by the surgeon in turn to tele-operate on the patient. If tele-operation is not compatible with the motion compensation algorithm developed here, the purpose of motion compensation is defeated. The tele-operation procedure requires the kinematics of both the master and slave and hence they form another important block in the flowchart. The implementation of tele-operation is described in Section VII.

IV. daVinci Surgical Robot

Fig. 2. daVinci surgical system showing all consoles

The daVinci Surgical System is a robotic surgical system made by the American company Intuitive Surgical. It is designed to facilitate complex surgery using a minimally invasive approach, and is controlled by a surgeon from a console. It has a master tool manipulator (MTM) which is used by the surgeon to control the patient side manipulator (PSM) which operates on the patient.

MTM is an over actuated system as it has 7 degrees of freedom. This over actuation is made keeping in mind the ergonomics for the surgeon and so that the whole system is stable when the needle is being inserted. If it were not over actuated, the tool could not be inserted keeping the other position of the arm the same.

PSM on the other hand is also a seven degrees of freedom, but the 7^{th} degree is the manipulation of the gripper tool and does not have a spatial importance. Hence, for the inverse kinematics, one can consider only the first six degrees of freedom. This calculation is shown in Section V-C. The interesting thing about the design of PSM is it has a remote center which is a fixed point in space with respect to the base frame and is independent of the joint angles. It is mechanically constrained to be fixed. This helps when the tool is used to make an incision and the remote center is kept fixed to the incision reducing the damage due to movement.

The daVinci robot is a open source hardware which allows research of this kind to be performed and documented for the advancement of the technology. A research toolkit, namely daVinci research toolkit is available online which includes all the design for parts and files used for simulation. Furthermore, there is a user guide which details the kinematics and positioning of joints. However, in this project since we are simulating the whole system in a simulation environments, our considerations are a bit different than what is provided and hence have recalculated the kinematics and is shown in the next section.

V. Kinematics of daVinci

This section presents the kinematics of the different consoles of the daVinci surgical robot. Here, the calculation used in this project is presented. The motion compensation requires the inverse kinematics of PSM, but to calculate inverse kinematics, the forward kinematics should be known. Moreover, the tele-operation requires the forward kinematics of the MTM. However, inverse kinematics for the MTM is not needed as the MTM does not have to controlled by PSM. It is controlled by the surgeon during surgery and hence only forward kinematics is needed to determine the displacement of its end effector which is the input for the PSM.

A. Forward Kinematics of MTM

MTM is a 7 degree of freedom arm, which has two yaw joints at the wrist considering the ergonomics of the surgeon. Denavit-Hartenberg (DH) parameters were used to formulate the forward kinematics of the MTM. Theses parameters have been presented in Table I. All joints in it are revolute joints. The manipulator is a combination of an articulated arm with a spherical wrist. The only difference is that the spherical wrist is attached on a moving platform.

TABLE I

DH parameters for Master Tool Manipulator

Frame	Joint name	a_i	α_i	d_i	θ_i
1	Outer Yaw	0	$-\pi/2$	0	$\pi/2 + q_1$
2	Outer Pitch 1	$-l_a$	0	0	$-\pi/2 + q_2$
3	Outer Pitch 2	$-l_f$	$\pi/2$	0	$\pi/2 + q_1$
4	Platform Yaw	0	$-\pi/2$	h	q_4
5	Wrist Pitch	0	$\pi/2$	0	q_5
6	Wrist Yaw	0	$\pi/2$	0	$\pi/2 + q_6$
7	Wrist Roll	0	0	0	q_7

B. Forward Kinematics of PSM

The Patient Side Manipulator on the other hand is a 6 Degree of freedom manipulator. It has one prismatic joint which is the insertion joint for the instrument with which the patient is operated on. All the other joints are revolute. However, there is a parallelogram link in it which makes the kinematics of it a bit non trivial.

The PSM is designed in such a way that it has a point which remains fixed irrespective of the joint angles. It is a fixed point in space, but its orientation can change. This fixed point proves to be useful when operating inside the body which is most often the case. The point is fixed to the incision made and makes sure that the incision does not widen, hence making it a minimal invasive surgery. This remote center mechanism is a mechanical constraint enforced due to a parallelogram link.

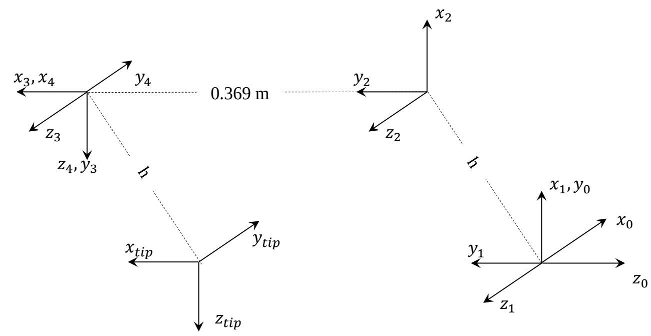

Fig. 3. Frame assignments for the parallelogram link in PSM

TABLE II

DH parameters for Patient Side Manipulator

Frame	Joint name	a_i	α_i	d_i	θ_i
1	Outer Yaw	$q_1 + \pi/2$	0	0	$-\pi/2$
2	Outer Pitch 1	q_2	h	π	π
3	Outer Pitch 2	$q_2 - \pi/2$	0	-0.369	π
4	Outer Pitch 3	q_2	0	0	$-\pi/2$
5	Insertion	0	$h + q_3$	0	0

Though only one joint of the parallelogram link is actuated and the rest are passive and mimic the joint angle, it is important to consider these joints in the DH parameter assignment. If they are not considered, the parallelogram link acts as a rigid link rendering the remote center mechanism useless. The frame assignment for the link is shown in Fig. 3. The DH parameter corresponding to these frames are shown in Table II. The position of the tip in frame F_0 is given by,

$$x = -d_3 cos(\theta_2) sin(\theta_1) \tag{1}$$

$$y = -d_3 cos(\theta_2) cos(\theta_1) \tag{2}$$

$$z = -d_3 sin(\theta_2) - 0.369 \tag{3}$$

Since in this project we only need the position of the tip of the needle of PSM, we show here only the DH parameters corresponding to calculating the position. There are 3 more joints other than the ones shown which are responsible for the roll, pitch and yaw of the needle. However, in motion compensation only the tip position is important and not its orientation. In addition, this information is sufficient to calculate the inverse kinematics, specifically inverse position kinematics as shown in the next subsection.

C. Inverse Kinematics of PSM

This section gives the inverse position kinematics of the PSM tip. Hence, it calculates for a given position of the tip, the yaw angle θ_1, the parallelogram link pitch angle θ_2, and the insertion length d_3. The insertion length is zero when the tip is at the remote center.

Solving the equations 1-3 for θ_1, θ_2, θ_3,

$$\theta_1 = atan2(x, y) \tag{4}$$

$$\theta_2 = atan2(1 - \sqrt{x^2 + y^2}, \sqrt{x^2 + y^2}) \tag{5}$$

$$d_3 = \frac{z + 0.369}{sin(\theta_2)} \tag{6}$$

VI. Motion Compensation Algorithm

The motion compensation algorithm developed during the course of this project is presented in this section. The main purpose of such an algorithm to be developed is to compensate for motion of the organs so that the instrument tip moves in sync with the organ. This makes the surgeries a lot more easier, especially when they are tele-operated. Currently, the surgeries performed on for instance beating heart require a lot of skill from the surgeon. The breathing has to be controlled to move in sync with the heart. Another approach is to run a pump in place of the heart and operate on the static heart. A comparison of both approaches can be found in [14]. Our aim is to mimic the effects of the second approach in terms of the operating conditions while keeping the advantages of the former.

For the project, we investigate our algorithm on a organ like movement of a platform oscillating with one degree of freedom. This platform was given a linear combination of hundred sine waves, frequency of which were ranging from $0\,hz$ to $5\,hz$ and with random amplitudes and phase differences. The result of this combination was a random wave generated which we knew nothing about. This is similar to the organ movements such as the eye when under topical anesthesia during phacoemulsification surgeries. There are some sudden jumps in the wave where amplitude goes very high or very low. This is the result of all individual sines adding up and increasing the amplitude.

Due to the short duration of the project, we were unable to incorporate a sensor in the simulation environment. However to simulate the measurements of the location of the platform, we added a random noise to the combination of sine waves generated as described above. A variance of 25% of the maximum amplitude was chosen for this random noise. This now was given as an input to the predictive filter such as Kalman filter.

The workings of Kalman filter are described in the Preliminary section and is used as the predictive filter to predict the next state of the platform based on the measurements. The state vector x_t chosen for the Kalman filter is the position of the platform and its velocity. This state follows all the requirements of the Kalman filter as it results in a linear relationship between the current and the next state. Since the measurements z_t too represents the position of the platform plus added variance, the state can be represented as

$$x_t = A_t x_{t-1} \tag{7}$$

$$x_t = C_t z_t \tag{8}$$

where $A_t = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$ and $C_t = \begin{bmatrix} 1 & 0 \end{bmatrix}$, $\Delta t = 0.1$. The Matrix Q_t in Algorithm 1 is the variance of the measurements and the matrix R_t is taken as,

$$R_t = \begin{bmatrix} \Delta_t \\ 1 \end{bmatrix} * 100 * \begin{bmatrix} \Delta_t & 1 \end{bmatrix} \tag{9}$$

This matrix was purely determined by a trial and error method and gave the least rms value. This matrix represents the uncertainty in the prediction update of the Kalman filter and the number 100 tells us that there is a high uncertainty. This is expected as the state is varying continuously and keeping a uncertainty makes the algorithm adjust to these changes faster, hence keeping a closer track of the actual state.

The Kalman filter algorithm outputs the next predicted state of the platform which includes the position and the velocity. For now, we chose to ignore the velocity even though it can be used if a velocity controller is used instead of position controller. Currently we have used a position controller which moves the system to a desired set of joint angles. The predicted position given by the algorithm is given to the inverse kinematics to get the joint angles. These joint angles are given to the position controller and the robot is moved.

The other approach of doing the motion compensation is by using the force feedback. Although this method may be more accurate in tracking the tissue movement, a contact has to be maintained at all points. Furthermore, there is delay in the feedback and the action done may cause more harm than only compensate. Due to the phase difference caused by the delay, the needle may be moving in a direction opposite to the tissue which is very dangerous. Hence, a predictive algorithm is surely needed which can reduce this delay.

This motion compensation was tested in the simulation environment using Gazebo and a new ROS package was written to implement it. This is described in the section VIII. The evaluation of the algorithm and its results when implemented in the simulation are discusses in the section IX.

VII. Teleoperation

Tele-operation is operation of a slave robot performing actions at a distance controlled by a master console. The novelty of daVinci surgical robot is it allows such tele-operation and still is very precise and accurate. One of the major goals of this project was to incorporate motion compensation in the normal teleoperation of surgery. The MTM being controlled by the surgeons gives the commands to PSM which is operating on the patient. The advantages of teleperation include reduced risk of surgeon getting exposed, better ergonomics for the surgeon and better precision during surgery.

The motion of the MTM made by the surgeon records the joint angles. Using forward kinematics of the MTM end effector, its displacement can be calculated. This displacement is scaled by a factor (0.2 in this project). This displacement is then added to the current end effector position of the PSm to get its final position. This final position can be given to the inverse kinematics and the corresponding joint angles are calculated. The controller then moves the PSM to these joint angles completing the tele-operation.

This teleoperation is differential with respect to PSM. This means that the PSM does not follow the exact joints of the MTM but it follows the path of the MTM. This makes the control more intuitive for the surgeon and hence is easier to operate with. We have implemented the tele-operation along with motion compensation and the results can be in seen in Section IX.

VIII. Simulation

A. Master Tool Manipulator (MTM)

The universal robot description format (URDF) of the MTM is available online on Github (https://github.com/WPI-AIM/dvrk-ros). In the repository, everything is written as a RViz package only. However, for our project we needed to simulate DaVinci in Gazebo as there is a need to simulate the dynamics of the motion. Hence, we modified the URDFs to include inertial parameters and transmission tags for the actuators.

Furthermore, a control file was written which launches the controllers (currently PID). This enables us to control the position of each of the joints via a C++ or python script. These controllers are called when the teleoperation is enabled and some inputs to the joints of the MTM are given. The teleoperation is run along with the motion compensation algorithm in the package made during this project. The package includes the kinematics of the whole system, motion compensation algorithm, code for tele-operation and movement of the platform simulating an organ.

B. Patient Side Manipulator (PSM)

Similar to MTM, the files for PSM are available open source but not configured for running Gazebo simulation. Although there are only six actuated joint angles, the robot description for the PSM has more joints. Some links like the edge of parallelogram is supposed to be a rigid link, whereas it has two components top and bottom both connected to the link 2 which pitches by two separate joints. These two joints have to be moved simultaneously so that the links do not detach with respect to each other. Furthermore, the parallelogram link has more joints than are actuated. These also have to follow the joint 2 during the robotic arm motion. Again, the controllers for all joints have to be written and the PID values tuned. These controllers are called after with the joint angles calculated by the inverse kinematics as the inputs.

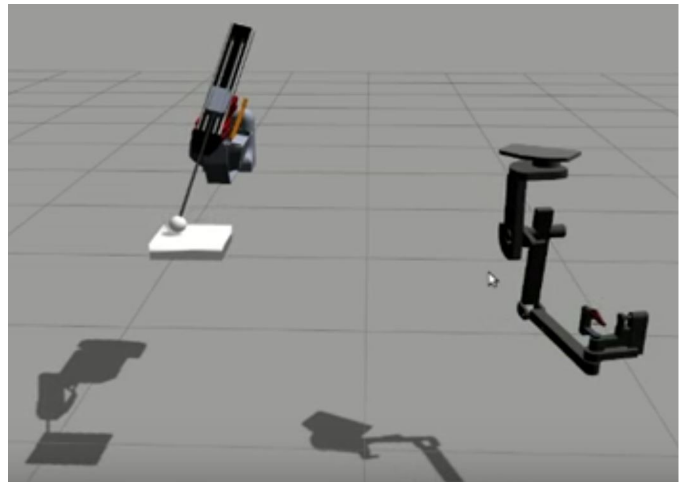

Fig. 4. Gazebo simulation environment showing the MTM, PSM and the platform representing an organ.

C. Simulating organ movement

To simulate organ movement, a cuboidal platform of size 0.2, 0.2, $0.05m$ was created. A separate URDF was written connecting this platform to the world using a prismatic joint. The actuation of this prismatic is given by the Fourier series as described in the Section VI. For this actuation too a position controller was written. The position of this prismatic joint was given a limit of $\pm 0.12\,m$ since the insertion can at most be of $0.24\,m$

The process of integrating the three models was a bit problematic. Individually each of the three robot model would spawn along with their respective controllers. However, while spawning all three together, the controllers failed to launch. The reason for this failure was that the *controller_manager* package used for position control required a *robot_description* parameter where the robot description was present. Since there were three robot descriptions, only the controllers for which the description was defined in the *robot_description* were loaded. A temporary fix to this problem that was employed in this project was to write a single URDF which includes all the three robot models and load the controllers on it. This fix works in this case, but may not suitable for other multi-body simulations.

Fig. 4 shows the Gazebo setup where all the three models are spawned with their controllers. We can see that the PSM is not at its home position and is controlled by the inputs given to the controllers.

IX. Results

The predictors were implemented using a linear combination of multiple sine and cosine functions with noise to model input from a real sensor. A predictor using Kalman filter with position, velocity as input was implemented. The results of the prediction by the kalman filter are as shown in the Fig. 6 in comparison to the actual motion of the organ. We can observe from the figure that the prediction

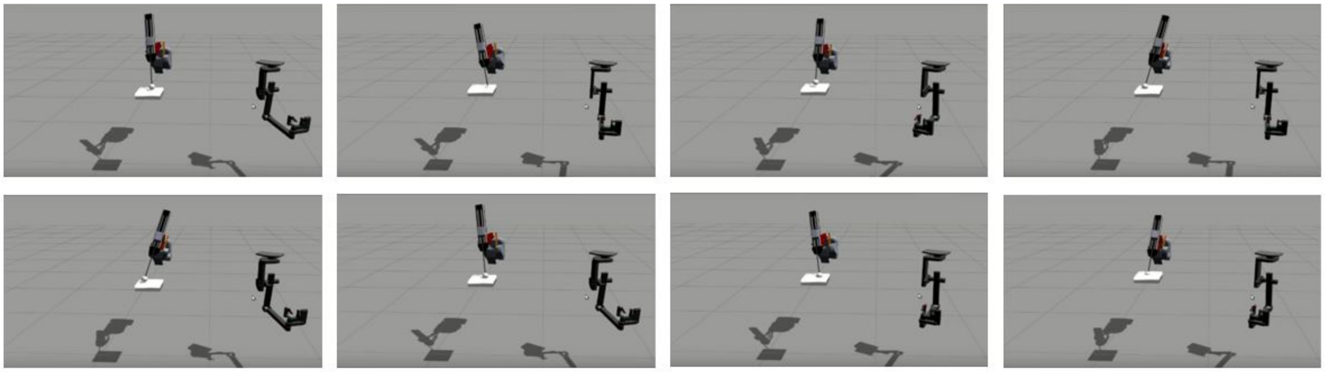

Fig. 5. Series of snapshots taken from the screen capture while the teleoperation was being simulated along-with motion compensation. The PSM moves according to the MTM motion but also keeps on compensating for the motion of the organ

tracks the desired motion albeit with an error at peaks and troughs.There is no apparent phase difference between the actual and predicted motions.This error can be reduced by using other predictive filters such as Extended Kalman Filter etc.This motion prediction algorithm will then be combined with the controller of PSM so that the joint angles are modified every instant to ensure that the endpoint of the PSM compensates for the motion of the organ. Moreover, it is also ensured that for any position and orientation of the end point of the PSM motion compensation is being done. This is the case in real life scenarios. Furthermore, the process of teleoperation is also simulated along with motion simulation by scaling down the motion of the MTM and reflecting it on the PSM.

Fig. 5 shows a series of snapshots. We can see that the PSM moves according to the movement of MTM and also compensates for the moving organ motion.

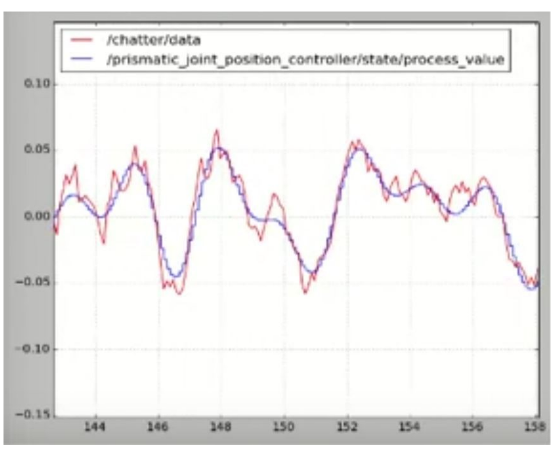

Fig. 6. Graph showing the prediction of Kalman filter. The blue line represents the position of the platform. The red line represents the predicted motion given as an input to the inverse kinematics of the PSM. The graph is plotted during the real time simulation in Gazebo.

X. Conclusion

In this project, motion compensation of the daVinci robot for micro-surgery has been simulated in the gazebo simulation environment. A detailed study of kinematics of the PSM and MTM has been carried out by which we could control the joint angles to incorporate motion of the organ. A kalman filter has been used to predict the motion of the organ which is being operated. Moreover a controller was designed to ensure that the PSM of the daVinci robot follows the predicted motion of the organ accurately. In this project rather than just simulating the motion compensation of the daVinci robot, teleoperation of the PSM based on the movement of the MTM is also simulated. Integrating motion compensation with teleoperation is an important step which is simulated as well. This makes sure that the surgeon does not have to worry about the motion of the moving organ while performing the surgery.

This project has more scope for improvement. While simulating the daVinci robot in the gazebo environment, the inertial parameters were taken randomly. So, in the future the simulation will be carried out with the exact inertia values of the robot for a more realistic simulation. Moreover, the PID controller needs to be tuned for accurate movement of the various joints in the PSM. In this project only kalman filter was used to predict the motion of the moving organ. In order to get better results, various other predictive filters like extended kalman filter could be used. Furthermore, one important future work is to integrate this with the existing daVinci hardware setup to test the feasibility in the real world.

References

[1] Schweikard, Achim, et al. ”Robotic motion compensation for respiratory movement during radiosurgery.” Computer Aided Surgery 5.4 (2000): 263-277.

[2] Yuen, S. G., Kettler, D. T., Novotny, P. M., Plowes, R. D., & Howe, R. D. (2009). Robotic motion compensation for beating heart intracardiac surgery. The International journal of robotics research, 28(10), 1355-1372.

[3] Ramrath, Lukas, et al. ”Prediction of respiratory motion with a multi-frequency based Extended Kalman Filter.” Proceedings of the 21st International Conference and Exhibition on Computer Assisted Radiology and Surgery (CARS?07). Vol. 21. 2007. 1

[4] Ortmaier, Tobias Johannes. Motion compensation in minimally invasive robotic surgery. Diss. Universitt Mnchen, 2003.

[5] Franke, T. J., Bebek, O., & Cavusoglu, M. C. (2008, May). Prediction of heartbeat motion with a generalized adaptive filter. In Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on (pp. 2916-2921). IEEE.

[6] Zarrouk, Z., Chemori, A., & Poignet, P. (2013, October). Force feedback control for compensation of physiological motions in beating heart surgery with real-time experiments. In 3rd International Conference on Systems and Control (pp. 956-961). IEEE.

[7] Riviere, Cameron N., Jacques Gangloff, and Michel de Mathelin. "Robotic compensation of biological motion to enhance surgical accuracy." PROCEEDINGS-IEEE 94.9 (2006): 1705.

[8] Cagneau, Barthelemy, et al. "Physiological motion compensation in robotized surgery using force feedback control." Proceedings 2007 IEEE International Conference on Robotics and Automation. IEEE, 2007.

[9] Richa, Rogrio, Antnio PL B, and Philippe Poignet. "Towards robust 3D visual tracking for motion compensation in beating heart surgery." Medical Image Analysis 15.3 (2011): 302-315.

[10] Kettler, Daniel T., et al. "An active motion compensation instrument for beating heart mitral valve surgery." 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2007.

[11] Richa, Rogrio, Philippe Poignet, and Chao Liu. "Efficient 3D tracking for motion compensation in beating heart surgery." International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer Berlin Heidelberg, 2008.

[12] Joseph, Lentin. Mastering ROS for robotics programming. Packt Publishing Ltd, 2015.

[13] Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. Probabilistic robotics. MIT press, 2005.

[14] Angelini, Gianni D., et al. "Early and midterm outcome after off-pump and on-pump surgery in Beating Heart Against Cardioplegic Arrest Studies (BHACAS 1 and 2): a pooled analysis of two randomised controlled trials." The Lancet 359.9313 (2002): 1194-1199.