Vision-based Control of a Soft Robot for Maskless Head and Neck Cancer Radiotherapy

Olalekan P. Ogunmolu[1], Xuejun Gu[2], Steve Jiang[2], and Nicholas R. Gans[1]

***Abstract*— This work presents an on-going investigation of the control of a pneumatic soft-robot actuator addressing accurate patient positioning systems in maskless head and neck cancer radiotherapy. We employ two RGB-D sensors in a sensor fusion scheme to better estimate a patient's head pitch motion. A system identification prediction error model is used to obtain a linear time invariant state space model. We then use the model to design a linear quadratic Gaussian feedback controller to manipulate the patient head position based on sensed head pitch motion. Experiments demonstrate the success of our approach.**

I. Introduction

This paper presents a continuation of our investigation of an image-guided soft robot patient positioning system for use in head and neck (H&N) cancer radiotherapy (RT). In 2014, over 1.6 million patients developed pharynx and oral cavity cancers in the United States, which led to over 580,000 deaths [1]. Typical H&N cancer treatment involves intensity-modulated radiotherapy (IMRT), which delivers high potent dose to tumors while simultaneously minimizing dose to adjacent critical organs such as spinal cord, parotids glands, and optical nerves. Typically, a patient lies on a 6-DOF movable treatment couch, and laser or image-guidance systems are used to ensure the patient is in the proper position.

IMRT requires accurate patient positioning. An examination of patient displacement and beam angle misalignment during IMRT showed errors as small as 3-mm caused 38% decrease in minimum target dose or 41% increase in the maximum spinal cord dose [2]. Image-guided radiotherapy (IGRT) has improved IMRT accuracy while reducing set-up times [3], [4], [5]. However, current IGRT practices focus on using images acquired before treatment to confirm beam placement [6]. The discomfort caused by head masks in prolonged IMRT treatment can increase patients voluntary and involuntary motion. Studies show that translational errors caused by patient motion can be larger than 6mm, and rotational errors can be as high as 2° [7]. Current motion-tracking systems, such as Cyberknife and Novalis are not compatible with conventional linear particle accelerators used at the majority of cancer centers. Moreover, these two systems are limited to assuming the patient's body is rigid during motion tracking and compensation. Recently, a robotic real-time surface image-guided positioning system was studied for feasibility in frameless and maskless cranial stereotactic radiosurgery [8]. While it achieved similar accuracy as the existing clinical methods, the system may not be suitable to IMRT due to the presence of mechanical and electrical parts in the path of the radiation beam.

Soft robot systems are deformable polymer enclosures with fluid-filled chambers that enable manipulation and locomotion tasks by a proportional control of the amount of fluid in the chamber [9], [10]. Their customizable, deformable nature and compliance make them suitable to biomedical applications as opposed to rigid and stiff mechanical robot components. They can also be made radiotransparent, which is necessary in IMRT.

The long term goal of our work is to address the non-rigid motion compensation during H&N RT. As we continue our initial investigation, we control one degree of freedom, raising or lowering of a generic patient's head, lying in a supine position, to a desired height above a table. The current system consists of a single inflatable air bladder (IAB), a mannequin head and a neck/torso motion simulator, two different Kinect RGB-D cameras to measure patient position, two current-controlled pneumatic valve actuators, and a National Instruments myRIO microcontroller. In this work, we extended and improve our previous work [11]. This paper contributes better vision tracking and localization methods via filtering and fusion of the two RGB-D estimates. We improve on the system identification of the soft-robot system and now incorporate an optimal control network. The result is a much improved motion control.

Section II of this paper briefly presents the design of the soft robot system. Section III discusses the computer vision algorithms to detect the patient's face and fusion of measurements from the RB-D images. Section IV presents results of system identification for the soft-robot system. Section V presents design of the linear quadratic Gaussian (LQG) controller, and Section VI presents several experiments to demonstrate the system.

II. Soft Robot Design Overview

The soft robot actuation mechanism combines an IAB (19" x 12") made of lightweight, durable and deformable polyester and PVC, two current-controlled proportional solenoid valves, and a pair of silicone rubber tubes (attached to a T-port connector at the orifice of the IAB) in order to convey air in/out of the IAB. A 1HP air compressor supplied regulated air at 30 psi to the inlet actuating valve, while an interconnection of a 60W micro-diaphragm pump and a valve removed air from the outlet terminal of the IAB. The RGB-D sensors are mounted directly above the head for raw head position and velocity measurements, while local Kalman filters (KFs) provide two estimates of the head position and velocity. The sensor estimates are aggregated using a track-to-track KF-based sensor fusion algorithm. We apply the

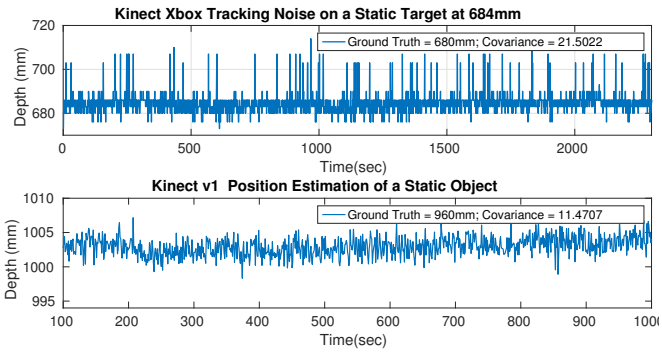

Fig. 1: Noise floor of Kinect Xbox Sensor vs. Kinect v2 Sensor

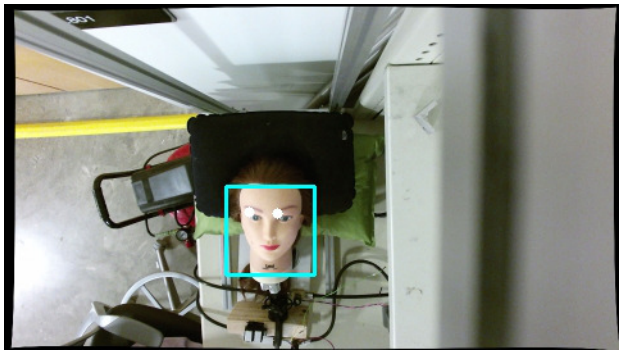

Fig. 2: Original colored image retrieved from the Kinect v2 Sensor.

fusion result in a new robust control law for the pneumatic actuator valves, thereby regulating air pressure within the IAB and moving the patient's head as desired. The real-time controller was deployed on a National Instruments myRIO embedded system running LabVIEW 2015. The LabVIEW algorithms were processed within a Windows 7 virtualbox running on the Ubuntu host workstation.

We use a Kinect Xbox 360, and a Kinect for Windows v2 sensor to estimate head position and velocity. The two sensors use different electronic perception technologies to determine distance of an object from the camera origin. They therefore have different lateral and range resolutions as well as different noise characteristics. Image processing for both cameras is executed on a 22GB RAM mobile workstation with Intel Core i7-4800MQ processor running 64-bit Ubuntu Trusty on a Linux 4.04 kernel.

III. Image-Based Patient Position Estimation

We perform recursive filter estimations of the RGB-D measurements and improve position estimates by using an additional sensor to better localize tracked features. We add the Kinect v2 sensor (henceforth called the v2 sensor), based on the time-of-flight (ToF) electronic perception principle. In ToF, light pulses illuminate a scene, and depth is calculated by determining the phase shift of the returned light signals. The active infra-red reduces the dependence on ambient lighting [12], and this sensor has a higher spatial depth resolution of 512×424 pixels at 30Hz interactive rate, compared to the Xbox's 320 × 240 pixels [13]. To minimize the noise due to the limited sensor resolution, the v2 has in-built noise improvement capabilities [14].

The v2 provides a higher depth-map accuracy and lower noise floor compared to Kinect for Xbox, as can be seen from Fig. 1, where the v2 exhibits a noise auto-covariance of $11.4707mm^2$ compared with $22.7057mm^2$ for the Xbox. Despite the improved performance of the v2, noise remains an issue, as is the case for every electronic perception system. To alleviate this, we employ a multisensor data fusion of both Kinect sensors' observations. We achieved this by local Kalman Filter estimates of each sensors observations, and we fuse the estimates via a variance-weighted multisensor Kalman filter fusion scheme described later in this section.

A. Face Detection and eye-feature tracking

We approached face detection using Haar Cascade Classifiers (HCC) [15]. HCC's are based on integral image representation, which allow for features evaluation while maintaining high detection rates. The features resemble Haar basis functions. A classifier is formed by choosing a small number of crucial features with AdaBoost, and a weighted sum of individual classifiers is used to construct a strong object detection classifier in a cascade manner. This increases the detector's speed by concentrating on areas within an image with high probability of features of interest.

A drawback of HCC's is the memory consumed on computing devices when searching through image pixels for specific regions of interest. Searching through a 640 × 480 pixels gray-scale image for specific features caused a 90% reduction in the frame rates of either sensor, when the algorithm is run on a CPU. To overcome this, both sensor's images were spatially down-sampled via linear interpolation before HCCs were applied. Face detection was performed on a single NVIDIA Quadro K1100M GPU. We retrieve each detected face from the GPU, and then detect eyes within detected faces using the same procedure.

To achieve robust detection, the minimum number of neighbors in each candidate rectangle feature was determined based on our experience. The search area within an image was chosen to be within the range of (5 × 5) pixels and (20 × 20) pixels. This gave us more than 90% face detection rate for both sensors. A similar approach was used for the eye classifier. The final implementation achieved a frame rate of 15Hz for each sensor running independently on the Linux host computer. Further improvement in frame rates is an avenue for future work.

B. Local Kalman Filters

From Fig. 1, we see that both RGB-D sensors suffer from notable associated noise, which is not suitable for our control requirement. To refine the observation, local Kalman Filter (KF) estimates for each sensor were computed to determine state estimates $\hat{\mathbf{x}}(i)$ that minimizes the mean-squared error to the true state $\mathbf{x}(i)$, given a measurement

sequence $z(1), \cdots, z(j)$, that is

$$\hat{\mathbf{x}}(i|j) = \arg \min_{\hat{\mathbf{x}}(i|j) \in \mathbb{R}^n} \mathbb{E}\{(\mathbf{x}(i) - \hat{\mathbf{x}})(\mathbf{x}(i) - \hat{\mathbf{x}})|z(1), \cdots, z(j)\}$$
$$\triangleq \mathbb{E}\{\mathbf{x}(i)|z(1), \cdots, z(j)\} \triangleq \mathbb{E}\{\mathbf{x}(i)|Z^j\} \quad (1)$$

where the obtained estimate is the expected value of the state at time i given observations up to time j. The covariance of the estimation error is given by

$$\mathbf{P}(i|j) \triangleq \mathbb{E}\{(\mathbf{x}(i) - \hat{\mathbf{x}}(i|j)(\mathbf{x}(i) - \hat{\mathbf{x}}(i|j)^T|Z^j\}. \quad (2)$$

Assuming the model of the state is common to both sensors, and denoting the distance from the v2 to the head as $d(k)$, we define $\mathbf{x}(k) = [d(k), \dot{d}(k)]^T \in \mathbb{R}^2$ as the state vector of interest, and let ΔT be the time between steps $k-1$ and k. The model state update equations are given by

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{B}_k \mathbf{u}_k + \mathbf{G}_k \mathbf{w}_k \quad (3)$$

where $\mathbf{F}(k) \in \mathbb{R}^{2\times 2}$ is the state transition matrix given by

$$\mathbf{F} = \begin{bmatrix} 1 & \Delta T \\ 0 & 1 \end{bmatrix} \quad (4)$$

$\mathbf{u}(k) \in \mathbb{R}^2$ is the control input, $\mathbf{B}(k)$ is the control input matrix that maps inputs to system states, $\mathbf{G}(k) \in \mathbb{R}^{2\times 2}$ process noise matrix, and $\mathbf{w}(k) \in \mathbb{R}^2$ is a random variable that models the state uncertainty. In the absence of inputs $\mathbf{B}_k \mathbf{u}_k = 0$, and the model becomes

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{G}_k \mathbf{w}_k \quad (5)$$

where $\mathbf{w}_k$ is the effect of an unknown input and $\mathbf{G}_k$ applies that effect to the state vector, $\mathbf{x}_k$. The process noise is assumed unknown and is modeled as uncontrolled forces causing an acceleration a_k in the head position (a_k is thus a scalar random variable with normal distribution, zero mean and standard deviation σ_a). We model this into (3) by setting $\mathbf{G}_k$ to identity and set $\mathbf{w}(k) \sim \mathcal{N}(0, \mathbf{Q}(k))$ where the covariance matrix $\mathbf{Q}(k)$ is set to a random walk sequence defined by $\mathbf{W}_k = [\frac{\Delta T^2}{2}, \Delta T]^T$. Therefore, we find that

$$\mathbf{Q} = \mathbf{W}\mathbf{W}^T {\sigma_a}^2 = \begin{bmatrix} \frac{\Delta T^4}{4} & \frac{\Delta T^3}{2} \\ \frac{\Delta T^3}{2} & \Delta T^2 \end{bmatrix} {\sigma_a}^2. \quad (6)$$

Denoting the head displacement at time k as measured by the Xbox and v2 as $z_1(k)$ and $z_2(k)$ respectively, the sensors' measurements were mapped to the v2 reference frame and modeled as

$$z_s = \mathbf{H}_s(k)\mathbf{x}(k) + v_s(k) \qquad s = 1, 2 \quad (7)$$

where $\mathbf{H}_s(k) = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$ maps the system's state space into the observed space, and $v_s(k) \in \mathbb{R}$ is a random variable that models the sensor error. We define $v_s(k)$ as a normally distributed random variable with zero mean and variance σ_{rs}^2. We assume the random sequences $v_1(k)$,$v_2(k)$, $\mathbf{w}(k)$ are independent and uncorrelated in time.

At each time step, k, each local KF's priori and posteriori estimates are computed through the typical prediction and update phases

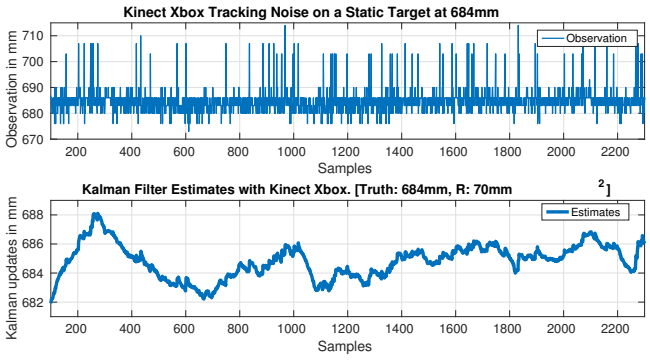

Fig. 3: KF results for the Xbox observation

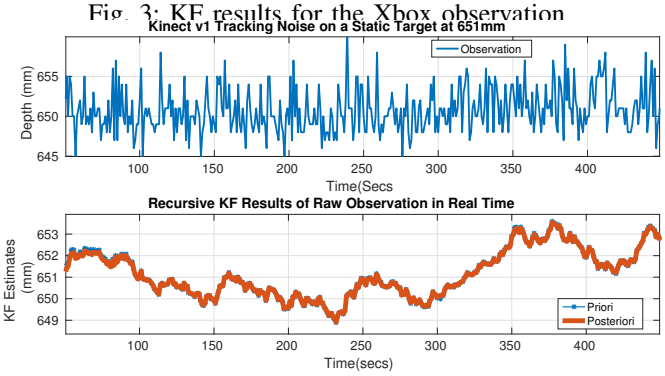

Fig. 4: KF results of Kinect v2's observation

Prediction Phase:

$$\hat{\mathbf{x}}(k|k-1) = \mathbf{F}(k)\hat{\mathbf{x}}(k-1|k-1) + \mathbf{B}(k)\mathbf{u}(k)$$
$$\mathbf{P}(k|k-1) = \mathbf{F}(k)\mathbf{P}(k-1|k-1)(\mathbf{F}(k))^T + \mathbf{Q}(k) \quad (8)$$

where $\hat{\mathbf{x}}(k|k-1)$ and $\mathbf{P}(k|k-1)$ are the state prediction vector and the prediction covariance matrix respectively.

Update Phase:

$$\mathbf{K}(k) = \mathbf{P}(k|k-1)\mathbf{H}(k)^T[\mathbf{H}(k)\mathbf{P}(k|k-1)\mathbf{H}(k)^T + \mathbf{R}(k)]^{-1}$$
$$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1))$$
$$\mathbf{P}(k|k) = (\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\mathbf{P}_(k|k-1) \quad (9)$$

where $\mathbf{K}_k$, $\hat{\mathbf{x}}_{k|k}$, and $\mathbf{P}_{k|k}$ are respectively the KF gain, posteriori state estimate and its state covariance matrix. In implementing the KF of (8) and (9), the variance of the process noise/signal noise of each local KF was informed by our knowledge of the physics of both sensors (electronic perception methods, range resolutions and examining each sensor's depth map to understand the data available to the filter), engineering judgment, and kinematics of the process model. We found these values sufficiently modeled the underlying process dynamics

$\sigma_a = 2000mm^2$; ${\sigma_{r1}}^2 = 70mm^2$for the Xbox, and ${\sigma_{r2}}^2 = 60mm^2$ for the Kinect v2 sensor.

Figs. 3 and 4 show the local filter estimate results of the observation from both the Kinect Xbox and v2 sensors post-filtering. The noise floor becomes noticeably reduced by each sensor after the KF filtering. The steady-state performance of both sensors include a reduction in the variance of the

observation sequence by 80.81%, while the Kinect v2 shows an improvement in noise rejection by almost 60% .

C. Data Fusion

Each local KF estimate was combined at a central fusion site to obtain a track-to-track fused global estimate. To communicate each estimate and associated covariance matrix, we create Unix FIFO special files (*i.e. named pipes*) on the kernel file system, write the estimates and covariance matrices to the pipes at each local site and retrieve the values at the central site.

Named pipes are low-level file I/O systems that can be shared by processes with different ancestry. During data exchange through a FIFO, the kernel forwards all data internally without having to write it to the file system. Since they exist within the kernel and the file system is just an entry serving as a reference point for the processes to access the pipe with a file system name, there is practically no delay in data communication.

Local tracks are generated at each sensor site according to (8), resulting in two local state predictions from the Kalman filters (3). At the central fusion site, we assume a state model common to both sensors given by (9) and adopt a variance-weighted average of each local track in the global track fusion algorithm [16]

$$\hat{\mathbf{x}}_F(k|k) = \mathbf{P}_F(k|k) \sum_{i=1}^{N} \left[{\mathbf{P}_s}^{-1}(k|k) \hat{\mathbf{x}}_s(k|k) \right]$$

$$\text{where } \mathbf{P}_F(k|k) = \left[\sum_{i=1}^{N} {\mathbf{P}_s}^{-1}(k|k) \right]^{-1}. \tag{10}$$

Fig. 5 shows the output of the fusion scheme compared against the single Kalman filters during a head-raising motion. The fusion of the local tracks produces better estimates, with improved signal to noise ratio. The fused estimate assigns more weight to the less noisy signal from Kinect v2. Through the implementation of the local tracks and a global track KF estimator, we improved the accuracy of the effective signal to be used in our control algorithm to no more than a standard deviation of 0.75mm from the true position of an object. The noise spikes in the fused tracks when the process state estimates are yet to converge as noticeable in Fig. 5 can be attributed to the noisy initialization of pixels in the sensors before they attain their steady state values. On average, it takes approximately 30 seconds for the pixel values in the Kinect sensor to reach their final steady state values [17]. This can be avoided by running the fusion algorithm for at least 2 minutes before the fused signal is used for any control purposes. The code for the multisensor fusion experiment is available on the git repos [18] and [19].

IV. Soft Robot System Identification

We approach the modeling procedure with an identification prediction error (PEM) approach, where we estimate a mathematical model, $G(t)$, based on the minimization of the

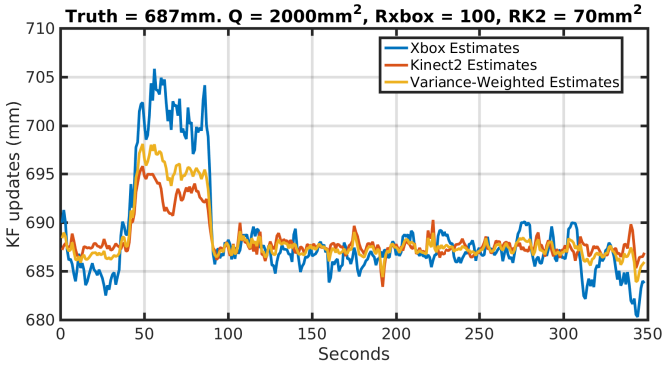

Fig. 5: Kalman filter Track-to-Track fusion of Kinect Xbox and v2's local tracks

sum of squared errors between estimates of the head height, $\hat{y(t)}$, and true head height, $y(t)$, from the fusion *i.e.*

$$G(t) = \arg \min_{\theta} V_N(\theta, Z^N)$$

$$\text{where } V_N(\theta, Z^N) = \sum_{k=1}^{\mathcal{K}} \sum_{i=1}^{n} \frac{1}{2} (\hat{y_i}(k) - y_i(k))^2. \tag{11}$$

$Z^N = \{u(1) \cdots u(N) \quad y(1) \cdots y(N)\}$ is the vector of past input and output (fused estimates) measurements over a bounded interval $[1, N]$ and θ is the greedy vector of parameters that approximate the model we seek to build. (11) is a special case of the least squares criterion.

A. Model Structure

Following Ljung's formulation in [20, §4.5], we pose the identification problem as determining the "best model" from a set of candidate model sets via an iterative approach that parametrizes the noncountable model sets smoothly over an area with the assumption that the underlying system is linear time-invariant. Here, our model structure is a differentiable mapping from a connected, compact subset $\mathcal{D}_{\mathfrak{M}}$ of $\mathcal{R}^d$ to a model set $\mathfrak{M}^*$, such that the gradients of the predictor functions are stable. This procedure is included in the MATLAB system identification toolbox, and since the method is well-documented in [21] we omit details.

External disturbances and stochastic variables are modeled as additive white noise sequence, $e(k)$, based on lagged inputs and outputs, and our objective is to estimate a stochastic state space model structure of the form

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{w}(k) \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) + \mathbf{v}(k) \end{aligned} \tag{12}$$

where the noise terms $\mathbf{w}(k)$ and $\mathbf{v}(k)$ compensate for the effect of disturbances beyond frequencies of interest to system dynamics and make the model robust to model uncertainties. Since u and y alone are measurable in our setup, the states $\mathbf{x}(k)$ are estimated and (12) becomes a linear regression problem, where all the unknown matrix entries are linear

combinations of the measured inputs and output variables. This can be written as

$$Y(k) = \Theta\Phi(k) + E(k) \tag{13}$$

where

$$Y(k) = \begin{bmatrix} \mathbf{x}(k+1) \\ \mathbf{y}(k) \end{bmatrix},\ \Theta = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}$$

$$\Phi(k) = \begin{bmatrix} \mathbf{x}(k) \\ \mathbf{u}(k) \end{bmatrix} \text{ and } E(k) = \begin{bmatrix} \mathbb{E}(w(k)) \\ \mathbb{E}(v(k)) \end{bmatrix}.$$

We assume the noise term is white in order to assure an unbiased model. The parameter estimation problem is then to estimate the $\mathbf{A}, \mathbf{B}, \mathbf{C},$ and $\mathbf{D}$ matrices by the linear least squares regression of (13) assuming no physical insight into the system (i.e. a black box model). $\mathbb{E}(\mathbf{w}(k))$ and $\mathbb{E}(\mathbf{v}(k))$ are estimated as a sampled sum of squared errors of the residuals.

B. Parameter Estimation

The input, $u(k)$, and output signals, $y(k)$, can be characterized by a linear difference equation of the form

$$\begin{aligned} y(k) \quad &= -a_1 y(k-1) - \cdots - a_{n_a} y(k-n_a) \\ &-b_1 u(k-1) - \cdots - b_{n_b} u(k-n_b) - e(k) \\ &-c_1 e(k-1) - c_{n_c} e(k-n_c) \end{aligned} \tag{14}$$

where $e(k)$ describes the equation error as a moving average of white noise, and we assume $e(k)$ has a bias-variance term λ. We can rearrange (14) using the vectors

$$\begin{aligned} \psi(k,\theta) = [&-y(k-1)\cdots - y(k-n_a)\ u(k-1)\cdots \\ &u(k-n_b),\ e(k-1,\theta),\cdots,e(k-n_c,\theta)]^T \end{aligned} \tag{15}$$

$$\theta = [-a_1,\cdots,-a_{n_a},-b_1,\cdots,-b_{n_b},-c_1,\cdots,-c_{n_c}]. \tag{16}$$

The adjustable parameters of (15) are elements of θ. In our prediction model, it is convenient to write (14) as a one-step-ahead predictor of the form

$$\hat{y}(k) = G(q,\theta)u(k) + H(q,\theta)\hat{e}(k) \tag{17}$$

$$\text{with} \quad G(q,\theta) = \frac{B(q)}{A(q)},\ H(q,\theta) = \frac{C(q)}{A(q)}$$

which is a complete autoregressive moving average with exogenous input (ARMAX) model. $G(q,\theta)$ represents the transfer function from input to output predictions, and $H(q,\theta)$ denotes the transfer function of prediction errors to the output model, $\hat{y}(k)$; q is the z-transform, z^{-1}, while $A(q)$, $B(q)$, and $C(q)$ are polynomials defined as

$$\begin{aligned} A(q) &= 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a}, \\ B(q) &= b_1 q^{-1} + \cdots + b_{n_b} q^{n_b}, \\ C(q) &= 1 + c_1 q^{-1} + \cdots + c_{n_c} q^{-n_c} \end{aligned} \tag{18}$$

[22]. The predictor turns out to be a linear filter of the form

$$\hat{y}(k|\theta) = W_y(q,\theta)y(k) + W_u(q,\theta)u(k) \tag{19}$$

$$\text{and } y(k) = G(q,\theta)u(k) + H(q,\theta)[y(k) - \hat{y}(k)] \tag{20}$$

where $H(q,\theta)$ is the noise model and $\hat{y}(k)$ above can be regarded as the one-step ahead predictor. After rearranging (19),we find that

$$W_y = 1 - H^{-1}(q,\theta) \text{ and } W_u(q,\theta) = G(q,\theta)H^{-1}(q,\theta)$$

such that the residual errors from (19) become

$$e(k) = [y(k) - G(q,\theta)u(k)]H^{-1}(q,\theta). \tag{21}$$

We can consider (21) as passing the prediction errors through a linear filter that allows extra freedom in dealing with non-momentary properties of the prediction errors. Since the model is that of a linear system, (21) satisfies our objective by approximating the prefilter with the choice of the noise model in (13).

The estimation problem is to predict the estimates, $\hat{y}(k|\theta)$ so that the errors, $\varepsilon(t,\theta) = \| y(t) - \hat{y}(t|\theta) \|_p$ are minimized by the choice of an appropriate p-norm criterion function, such as the mean squared error proposed in (11).

1) Input Signal Design: The input signal choice for a system identification experiment will determine a system's operating point and model accuracy. Therefore, the input should be rich enough to excite a system and force it to show properties needed for the model's purpose. For the model to be informative across all the desired frequency range, a periodic, persistently exciting uniform Gaussian White noise (UGWN) signal with clipped amplitudes corresponding to the bandwidth of the valves was designed offline, and its frequency spectrum analyzed to ensure it had as small a crest factor as possible (since the asymptotic properties of the model will be mostly influenced by the spectrum rather than the waveform's time-series shape). Gaussian White Noise signals (GWN) and Pseudo-Random Binary Signals (PRBS) are well-known to achieve virtually any signal spectrum without very narrow pass bands. Therefore, pseudo-random uniform white noise sequences were generated using the very-long cycle random number generator algorithm. Given that the probability density function, $f(x)$, of the uniformly distributed uniform white noise is

$$\begin{aligned} f(x) &= \frac{1}{2}A \qquad \text{if } x < |A| \text{ and} \\ u(x) &= 0 \qquad \text{if } x > |A| \end{aligned} \tag{22}$$

where A is the amplitude. The expected mean, μ, and the expected standard deviation, σ of the sequence are [23]

$$\mu = \mathbb{E}(x) = 0, \qquad \sigma = [\mathbb{E}\{(x-\mu)\}]^{\frac{1}{2}} = \frac{A}{\sqrt{3}}. \tag{23}$$

The spectrum of the resulting signal in Fig. 6 gives good signal power, which nicely relates to the bandwidth of the pneumatic valves and achieves virtually all signal spectrum with little narrow pass bands.

We therefore use the signal of (22) to model the desired asymptotic estimates, $\hat{y}(t)$, of (17). We sampled $y(t)$, the fused measurement described in (III-C) well-above the system's Nyquist frequencyand acquired enough samples to make Z^N asymptotically approach $\hat{\theta}_N$ as $N \to \infty$. The data collection procedure closely follows that described in

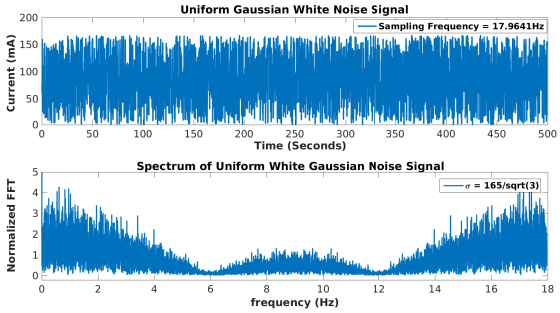

Fig. 6: Time/Frequency-Domain Properties of the Input Signal

our previous paper and we refer readers to [11, §IV.A] for a more detailed treatment.

The collected data was separated in a 60:40% ratio for training and testing purposes, respectively, to assure a training model that generalizes well.

2) State Space Realization: If we define

$$\hat{Y}_r(k) = [\hat{y}(k|k-1), \cdots, \hat{y}(k+r-1)|k-1]^T$$
$$\hat{Y} = [\hat{Y}_r(1) \cdots \hat{Y}(N)],$$

it follows that 1) as $N \to \infty$, there are n-th order minimal state space descriptions of the system if and only if the rank of the matrix of prediction vectors, $\hat{Y}$, is equal to n for all $r \geq n$; and 2) the state vector of any minimal realization in innovations can be chosen as linear combinations of $\hat{Y}_r$ that form a row basis for $\hat{Y}$, i.e.,

$$x(t) = L\hat{Y}_r(k)$$

with L being an $n \times pr$ matrix (p is the dimension of $y(k)$) [20, §7.3]. The true prediction is given by (17) with innovations $e(j)$ written as a linear combination of past input-output data. The predictor can thus be expressed as a linear function of $u(i)$, $y(i)$, $i \leq k-1$. In practice, the predictor is approximated so that it depends on a finite amount past data such as s_1 past outputs and s_2 past inputs of the form

$$\hat{y}(k|k-1) = \alpha_1 y(k-1) + \cdots + \alpha_{s_1} y(k-s_1) + \beta_1 u(k-1) + \cdots + \beta_{s_2} u(k-s_2). \quad (24)$$

Piping the identification data through the MATLAB function 'ssest' and testing various model orders based on the ranking of singular values of the Hankel matrix of input-output measurements [20], we obtained the results listed in Table I on training and testing dataset. The MATLAB system identification script is provided on a github repo [24] and contains the dataset used for the experiment. The model set above exhibit a high-fit of estimate to fed data with generally good mean-square errors and final prediction errors for a control experiment. With increasing model order starting from 4, we see that the fits start reaching convergence, as the mean-square errors and final prediction errors become constant. In the frequency-domain, this is the equivalent to having pole-zero cancellations for higher-order models. We therefore conclude there is no useful properties a higher-order model could predict beyond an order of 8. The second-order model sufficiently approximates the system and is not significantly outperformed by the higher order models –which would contribute higher complexity to the control design. We therefore pick the 2nd order state space model (12) as

TABLE I: Model estimates

Data Type	Expts	MO[1]	MSE[2]	Fit (%)	FPE [3]
Training	i	2	0.001437	97.64	0.001438
	ii	4	0.001454	97.62	0.00145584
	iii	6	0.001333	97.72	0.001336
	iv	8	0.001298	97.76	0.001298
Testing	i	2	0.000963	98.47	0.000964
	ii	4	0.0008574	98.56	0.008594
	iii	6	0.000846	98.57	0.000849
	iv	8	0.000843	98.57	0.000848

$$\mathbf{x}(k+Ts) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{K}\mathbf{e}(k)$$
$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) + \mathbf{e}(k) \quad (25)$$

where Ts is the sampling period, $\mathbf{e}(k)$ is the modeled zero-mean Gaussian white noise with non-zero variance,

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -0.9883 & 1.988 \end{bmatrix}, \ \mathbf{B} = \begin{bmatrix} -3.03e-07 \\ -4.254e-07 \end{bmatrix}$$
$$\mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \ D = 0, \text{and } \mathbf{K} = \begin{bmatrix} 0.9253 & 0.9604 \end{bmatrix}^T. \quad (26)$$

The pair (A, B) is stabilizable and the pair (A, C) is detectable.

V. LQG Control

We employ a LQG controller and estimator to minimize the following cost function subject to the state equation (26)

$$J = \sum_{k=0}^{\mathcal{K}} x^T(k)\, Q\, x(k) + R\, u(k)^T\, u(k) + 2x(k)^T\, N\, u(k) \quad (27)$$

where $\mathcal{K}$ is the terminal sampling instant, Q is a symmetric, positive semi-definite matrix that weights the n-states of the A matrix, N specifies a matrix of appropriate dimensions that penalizes the cross-product between the input and state vectors, while R is a symmetric, positive definite weighting matrix on the control vector u. The quadratic cost function in (27) allows us to find an analytical solution (controller sequence) to the minimization of J over the prediction horizon, n_y

$$\Delta u = \arg\min_{\Delta u} J \quad (28)$$

where Δu is the future control sequence and the first element in the sequence is used in the control law at every time instant. We model additive white noise disturbances into

[1]Model Order
[2]Mean Squared Error (mm^2).
[3]Akaike Final Prediction Error ([20, Secs 7.4 and 16.4]).

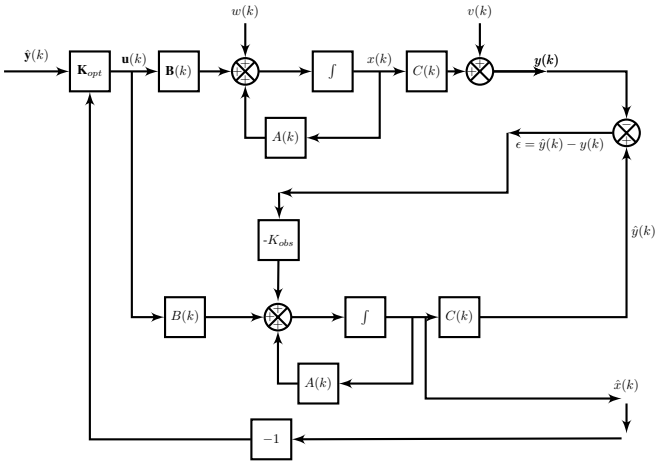

Fig. 7: Full Linear Quadratic Gaussian Plant Estimator

the discrete estimator's states; therefore the optimization problem becomes a stochastic optimization problem that must be solved.

The separation theorem ensures that we can construct a state estimator which asymptotically tracks the internal states from observed outputs, $y(k)$, using the algebraic Riccati equation given as

$$A^T PA\text{-}(A^T PB\text{+}N)(R\text{+}B^T PB)^{-1}(B^T PA\text{+}N)\text{+}Q. \quad (29)$$

where P is an unknown $n \times n$ symmetric matrix and A, B, Q, and R are known coefficient matrices as in (26) and (27). We find an optimal control law by solving the minimization of the LQ problem, (27) which we then feed into the states.

In practice, it is a good idea to start with an identity matrix, Q, a zero penalty matrix, N, and tune R till one obtains convergence by the state estimator. The following optimal values were used after a heuristic search

$$Q = \begin{bmatrix} 1.0566 & 0 \\ 0 & 1.0566 \end{bmatrix}, \; R = \begin{bmatrix} 0.058006 \end{bmatrix}. \quad (30)$$

We construct a full online estimator for the identified plant as in Fig. 7, whereby the noise processes are assumed to be independent, white, Gaussian, of zero mean and known covariances. The optimal controller gains, K_{opt}, are determined from the equation

$$K_{opt} = R^{-1}(B^T P + N^T) \quad (31)$$

[25] where P is the solution to the algebraic Riccati equation (29) and $\mathbb{E}[w(k)w'(\tau)] = R(k)\delta(k-\tau)$. Therefore, the online optimal estimate, $\hat{x}(k+1)$ of $x(k)$ is

$$\hat{x}(k+1) = A(k)\hat{x}(k) + K_{lqg}\left[C(k)\hat{x}(k) - y(k)\right] \quad (32)$$

where $\hat{x}(k_0) = \mathbb{E}\left[x(k_0)\right]$ The observer is equivalent to a discrete stochastic Kalman filter that estimates the optimal state $\hat{x}(k|k)$ as shown in Fig. 7. The estimator equations are similar to equations (8) and (9) and the online, unbiased estimate is

$$\begin{aligned} \hat{x}(k+1) &= A(k)\hat{x}(k) - K_{obs}[\hat{y}(k) - y(k)] + B(k)u(k) \\ \hat{y}(k) &= C(k)\hat{x}(k) \end{aligned} \quad (33)$$

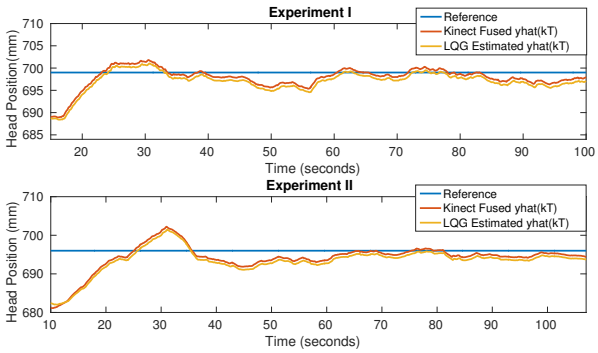

Fig. 8: LQG Controller on Manikin Head

$\Longrightarrow$

$$\begin{aligned} \hat{x}(k+1) = A(k)\hat{x}(k) - K_{obs}[C(k)\hat{x}(k) - y(k)] \\ + B(k)u(k). \end{aligned} \quad (34)$$

Through heuristics, we found the following variances of the online estimator to be useful:

$$Q_e = \begin{bmatrix} 0.4511 & 0 \\ 0 & 0.4511 \end{bmatrix}, \; R_e = \begin{bmatrix} 0.01 \end{bmatrix}$$

VI. Experimental Results and Discussion

The control network was implemented on an NI-myRIO running LabVIEW 2015. We initialized the Kinect sensors to allow for all pixels within the depth cameras to reach steady state under ambient light. We performed multiple experiments to evaluate the developed state space model of IV-B and LQG controller of V. [1]. The input variable is the current that excites the valve, which in turn actuates the bladder; the head moves in response to bladder actuation. The fused estimate of the Kinect sensors are used to estimate the real-time head pitch motion as described in III-C; this is in turn used in a feedback to the LQG controller.

Fig. 8 shows the results from a constant reference trajectory, which the head is meant to track. We notice a settling time of approximately 24 seconds before we reach steady state. The delay arises from our design requirements and is not a drawback in clinical trajectory tracking where we must ensure smooth head motion to desired target. It is also seen that the controller exhibits relatively smooth tracking within a 1.5 mm standard deviation over time after a relative overshoot of 5mm in bottom graph of Fig. 8. The overshoot can be explained by the estimator's search for a steady state region based on the time it takes for the pixel values of the sensors to reach steady state. The controller tracks the reference to within $\pm 2mm$.

However, we noticed an inconsistency at certain operating ranges in the current LTI model. The applied current based on fusion feedback occasionally reaches a steady state error, as can be seen from Fig. 9. We conjecture this is due to an unmodeled nonlinearity at the inlet valve that maps input currents to system states. To better approximate the

[1]The LabVIEW identification and control codes are available on the git repo [26].

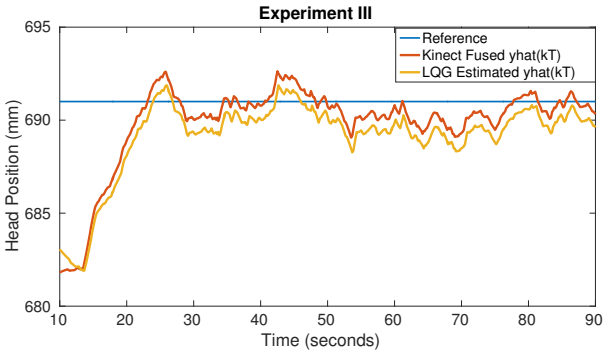

Fig. 9: LQG Controller on Manikin Head

nonlinearity from input to output, we will investigate using a Hammerstein block-structured model that better approximates the nonlinearity from inputs to states and states to output of the system.

The fusion algorithm proved useful to cancel jitter in the depth measurement of the sensor over our previous results, but it falls short of the 1mm accuracy in head and neck cancer RT. Having established proof of concept in this investigation, we will begin investigation of better head localization by using sophisticated motion capture systems or laser scanners, such as those actively employed in IGRT.

VII. Summary

We have presented a continuation of or initial investigation into a pneumatically-driven soft robot system for head and neck radiotherapy. Measurements from two Kinect RGB-D cameras are fused to refine the accuracy of observations. System identification on the soft robot was combined with LQG controller design to provide an optimal controller. Experiments showed we could actuate the patient head within 2.5mm accuracy. Future work will investigate accurate multi-axis positioning using better 3D sensor, control of multiple IABs, and nonlinear modeling or model free-methods to overcome limitations of the LTI model.

References

[1] R. Siegel *et al.*, "Cancer statistics," *CA: A Cancer Journal for Clinicians*, vol. 64, no. 1, pp. 9–29, 2014.

[2] L. Xing, "Dosimetric effects of patient displacement and collimator and gantry angle misalignment on intensity modulated radiation therapy," *Radiother Oncol*, vol. 56, no. 1, pp. 97–108, 2000.

[3] T. Takakura *et al.*, *The geometric accuracy of frameless stereotactic radiosurgery using a 6D robotic couch system*. Phys Med Biol, 2010.

[4] P. H. Ahn *et al.*, "Random positional variation among the skull, mandible, and cervical spine with treatment progression during head-and-neck radiotherapy," *Int J Radiat Oncol Biol Phys*, vol. 73, no. 2, pp. 626–33, 2009.

[5] D. Robb *et al.*, "Assessing the efficiency and consistency of daily image-guided radiation therapy in a modern radiotherapy centre," *Journal of Medical Imaging and Radiation Sciences*, 2013.

[6] D. Jaffray, "Image-guided radiotherapy: from current concept to future perspectives," *Nat Rev Clin Oncol*, vol. 9, no. 12, pp. 688–699, 2012.

[7] H. Kang, D. M. Lovelock, E. D. Yorke, S. Kriminiski, N. Lee, and H. I. Amols, "Accurate positioning for head and neck cancer patients using 2d and 3d image guidance," *J Appl Clin Med Phys*, vol. 12, no. 1, p. 3270, 2011.

[8] L. I. C. T. P. J. D. Lawson and S. B. Jiang, "Frame-less and mask-less cranial stereotactic radiosurgery: a feasibility study," *Phys. Med. Biol.*, vol. 55, pp. 1863–1873, 2010.

[9] C. Laschi *et al.*, "Soft robot arm inspired by the octopus," *Advanced Robotics*, vol. 26, no. 7, pp. 709–727, 2012.

[10] P. Maeder-York, T. Clites, E. Boggs, R. Neff, P. Polygerinos, ó. Holland, L. Stirling, K. Galloway, C. Wee, and C. Walsh, "Biologically inspired soft robot for thumb rehabilitation," *Journal of Medical Devices*, vol. 8, no. 2, p. 020933, 2014.

[11] O. Ogunmolu, X. Gu, S. Jiang, and N. Gans, "A Real-Time Soft Robotic Patient Positioning System for Maskless Head-and-Neck Cancer Radiotherapy: An Initial Investigation," in *IEEE International Conference on Automation Science and Engineering*, Gothenburg, Sweden, Aug 2015.

[12] (2015) Kinect Hardware. [Online]. Available: https://dev.windows.com/en-us/kinect/hardware

[13] (2015) Constants. [Online]. Available: https://msdn.microsoft.com/en-us/library/hh855368

[14] H. Gokturk, S.B. Yalcin and C. Bamji, "A Time-of-Flight Depth sensor - System Description, Issues and Solutions," in *2004 Conference on Computer Vision and Pattern Recognition Workshop*, 2004.

[15] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," pp. 511–518.

[16] H. Durrant-Whyte. (2001, Jan) Introduction to Estimation and the Kalman Filter. Accessed Aug 05, 2015. [Online]. Available: http://www.dynsyslab.org/archive/RecEst2010/www.idsc.ethz.ch/Courses/Archives/Recursive_Estimation/recursive_filtering_2010/EstimationNotes.pdf

[17] Andersen *et al.* (2012) Kinect Depth Sensor Evaluation for Computer Vision Applications. Accessed on Feb 23, 2016. [Online]. Available: http://eng.au.dk/fileadmin/DJF/ENG/PDF-filer/Tekniske_rapporter/Technical_Report_ECE-TR-6-samlet.pdf

[18] O. Ogunmolu. (2015) Face Tracker with the Xbox Sensor. Accessed on Feb. 24, 2016. [Online]. Available: https://github.com/lakehanne/xbox_tracker

[19] ——. (2015) IAI˙Kinect2. Accessed on Feb. 24, 2016. [Online]. Available: https://github.com/lakehanne/iai_kinect2

[20] L. Ljung, *System Identification Theory for the User*, 2nd ed. Upper Saddle River, NJ, USA.: Prentice Hall, 1999.

[21] L. Llung, *System Identification Toolbox^{TM} User's Guide R2014b*, Natick, MA, 2014.

[22] S. Billings, *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. John Wiley and Sons,Ltd, 2013, vol. 39, no. 3.

[23] N. Instruments. (2011) Uniform white noise vi - labview 2011. Accessed on March 26, 2016. [Online]. Available: http://zone.ni.com/reference/en-XX/help/371361H-01/lvanls/uniform_white_noise/

[24] O. Ogunmolu. (2015) Black Box Identification for Control. Accessed on Feb. 24, 2016. [Online]. Available: https://github.com/SeRViCE-Lab/Matlab-Files/blob/master/ident_data/Filtered%20GWN/carimaFWGN.m

[25] B. Anderson and J. Moore, *Optimal Control: Linear Quadratic Methods*. Prentice Hall, Englewood Cliffs, New Jersey 07632, 1990.

[26] O. Ogunmolu. (2015) RAL-Codes. Accessed on Feb 24, 2015. [Online]. Available: https://github.com/SeRViCE-Lab/RAL-Codes/blob/master/LQG%20Design/Soft_Robot_Model_ffwd.vi