Activity-Aware Recovery from Network Communication Loss in Teleoperated Robotic Surgery

Seyed Hamid Reza Roodabeh[1], Homa Alemzadeh[1]

***Abstract*—While network delay in teleoperated surgery has been studied, the critical safety risks of severe packet loss and total communication failures, which can cause unintended, potentially harmful robot movements, remain largely unaddressed. To bridge this gap, we introduce an activity-aware, dual-mode shared control framework that ensures operational continuity and safety. Our primary contribution is an intelligent recovery system that switches between two modes: (i) for short-term packet loss, a context-aware Transformer model accurately predicts surgeon commands, (ii) for prolonged communication failure, the system transitions to an autonomous mode and uses a library of specialized Dynamic Movement Primitives (DMP) to safely complete the current surgical subtask. Validated on a Peg Transfer (PT) task, our framework demonstrates significant improvements over existing methods. Under severe 50% packet loss, our adaptive switching and short-term recovery reduces trajectory deviation from nearly 26 mm to 3.62 mm, outperforming standard filtering methods with errors exceeding 6 mm. Most significantly, the integrated end-to-end framework improves trajectory accuracy by up to 60% compared to using either recovery method in isolation, proving the synergistic benefit of our dual-mode design for safely handling diverse network disruptions.**

***Index Terms*—Robotic Surgery, Teleoperation, Fault-Tolerant Control, Trajectory Planning**

I. INTRODUCTION

Robot-assisted Minimally Invasive Surgery (RAMIS) systems have become widely adopted due to their significant benefits, including enhanced precision and reduced patient hospitalization and recovery times [1]. Teleoperated surgical robots in particular, provide exceptional instrument guidance accuracy and advanced visualization capabilities. The da Vinci Surgical System is the most prominent example, with thousands of units deployed across the globe and nearly 17 million procedures successfully performed to date [2].

Teleoperation holds considerable promise in robotic surgery to enable remote procedures that can address the shortage of medical professionals and provide timely care in challenging and extreme environments, such as rural and underserved areas [3], battlefields [4], deep sea [5], and outer space [6]. However, several persistent technical challenges impede the real-world realization of remote surgery. Network reliability and security, in particular, have been identified as significant factors negatively impacting surgeon performance and patient outcomes [7], [8], [9]. Network issues can result in suboptimal operator performance, reduced precision, increased fatigue, and potentially severe harm to patients [10], [11]. The challenges of unreliable connectivity and limited bandwidth are especially pronounced in rural areas and battlefield conditions, where the risk of delay and jitter, severe packet loss or complete communication failure restricts the safe and dependable application of teleoperation.

Previous research has explored mitigating the adverse effects of network delays in teleoperated surgery [12], [13], but the problem of *severe, intermittent packet loss* or *total communication loss* has not been rigorously addressed. This issue threatens system availability and can lead to unintended instrument motions or deviation from the commanded trajectory, potentially causing patient harm. Different approaches such as predictive filtering [14], [15], multi-path connection [16] and passivity-based controllers [17] have been proposed to mitigate these issues. However, these methods often fail to consider the operational context necessary to execute robotic actions that should align with surgeon's higher-level intent and surgical task knowledge. This limits their applicability to shorter-term recovery scenarios and tasks that do not involve complex structures or faster-paced intricate instrument maneuvers, increasing the possibility of unwanted or erroneous movements. In addition, mitigation strategies for robotic telesurgery in network environments with co-occuring transient packet loss and total communication failure have not been studied before.

We propose a shared control-based *activity-aware recovery framework* designed to safeguard teleoperated surgical robots against *network packet drops* and *communication loss*. Our approach integrates surgical procedure knowledge, low-level activity recognition and autonomous planning and execution of primitive robotic movements, to design a safety engine that resides on the slave side of the teleoperated robotic system to detect and *automatically recover from network disruptions*. Unlike previous work, which relied on the limited fidelity of simulated environments or simplistic filtering methods for trajectory prediction [12], [15], our framework employs a *dual-mode autonomous recovery* procedure that accurately predicts surgeon commands, *in both short-term and long-term time horizons*, based on the kinematic measurements and video frames from the robot, and ensures continuous robot operations within safe operational states. Our method ensures that the system reaches a safe state even if communication loss extends beyond short transient interruptions studied previously [15], by switching from short-term to long-term recovery.

The main contributions of this work are as follows:

- Development of an autonomous shared-control recovery framework for real-time detection and mitigation of network packet drops and communication loss in teleoperated surgery.

[1]Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22903 USA. {ydq9ag,ha4d}@virginia.edu.

- Design of a dual-mode, adaptive recovery controller and activity-aware trajectory generation methods based on dynamic movement primitives (DMPs) and a Transformer [18] model for autonomously executing surgical actions (movement primitives) in the event of short-term and long-term network disruptions.
- Comprehensive experimental evaluation of the proposed recovery framework by simulating realistic communication loss scenarios on the pre-recorded PT task trials from two datasets: the publicly-available Dexterous Surgical Skills Transfer (DESK) dataset [19], and a dataset collected using the Raven surgical platform [20].

II. Background

A. Surgical Process Model

Earlier studies have structured surgical procedures using a hierarchical model [21] that includes steps, phases, tasks, gestures, and basic movement primitives (MPs), also called movement primitives [22]. Gestures are characterized as deliberate actions with specific semantic meaning, often involving particular instruments and objects/tissues. The framework in [23] breaks down surgical gestures into sequences of elementary instrument movements or MPs that cover fundamental actions like touching, pushing, pulling, and grasping. As such, MPs are atomic action blocks that can be combined to generate a large variety of complex surgical actions, making them a suitable model for highly generalizable trajectory generation.

Figure 1 shows the grammar graph of the PT task, a dry lab training routine from the Fundamentals of Laparoscopic Surgery (FLS) [24], which we use as our case study. This task involves transferring blocks across a pegboard using two graspers and is modeled using six unique MPs. Each MP is encoded as an action triplet, consisting of an action verb (e.g. Grasp), a surgical tool (e.g., grasper on the left arm), and the object with which the tool interacts (e.g., Peg). The left and right graspers are abbreviated with 'L' and 'R'. The gestures, MPs, and their average durations are shown in Table I. This task involves intricate movements that require high levels of precision and dexterity to handle small objects in tight spaces.

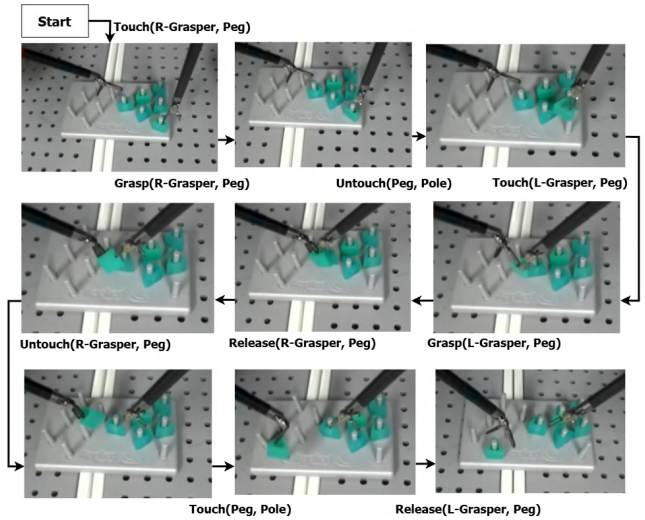

Fig. 1: Grammar Graph of the PT task.

TABLE I: Gestures and MPs in the PT Task.

Gesture	Description	Movement Primitives	Avg. Length (s)
S1	Approach Peg	**Touch**(L-Grasper, Peg)	4.4
S2	Align and Grasp	**Grasp**(L-Grasper, Peg)	1.22
S3	Lift Peg	**Untouch**(Peg, Pole1)	1.77
S4	Transfer Peg - Get Together	**Touch**(R-Grasper, Peg)	4.52
S5	Transfer peg - exchange	**Grasp**(R-Grasper, Peg)	1.93
		Release(L-Grasper, Peg)	0.91
		Untouch(L-Grasper, Peg)	1.11
S6	Approach Pole	**Touch**(Peg, Pole2)	5.88
S7	Align and place	**Release**(R-Grasper, Peg)	0.33

B. Surgical Task Automation and Shared Control

Automated task execution is often a core component in autonomous recovery frameworks. Automating surgical subtasks have been explored for object transfer [25], suturing [26], tissue retraction [27] and shunt insertion [28], among others. Although these works have achieved success in performing the underlying task autonomously, they do not consider a teleoperation setting that is prone to network interruption or connection loss. An autonomous task planning framework with situation awareness is introduced in [29], where executional errors were detected during the execution of fine-grained actions and replanning was performed in the case of errors. However, this work did not account for the system's ability to recover from communication errors in teleoperation.

In addition to task autonomy, the transition between autonomous and manual modes is also of great importance in achieving safe and effective fault recovery. Intelligent mode switching and shared control enhance human-robot collaboration by enabling flexible transitions between manual and autonomous control, especially under delayed or lossy communication channels. Deep reinforcement learning-based methods use user intention recognition and reward optimization [30] for optimal switching. Shared control strategies include confidence-based input blending [31], virtual fixtures for guided motion [32], model predictive control [33] and learning from demonstration [34]. In this work, we propose a simple yet effective and efficient way of adaptively switching between human and autonomy for networks prone to packet drop and communication loss, which can perform under real-time constraints of robotic teleoperated surgery.

C. Dynamic Movement Primitives

DMPs are a framework designed to encode and reproduce complex trajectories learned from human demonstrations. DMPs encode a motion trajectory (e.g., reaching or grasping) using a set of nonlinear differential equations with gain terms for ensuring stability and learned forcing terms capturing the motion shape. They offer robustness to scaling, handling movements of different magnitudes and invariance to rotation [35]. DMPs have been successfully applied to automating robotic manipulation tasks, including object transfer [29] and suturing [36]. Surgical robotics datasets have unique properties, such as multiple demonstrations, varied MP types, critical orientation data, and both successful and failed examples, which can be used for learning DMPs with different properties and constraints. We evaluate several state-of-the-art DMP formulations to build an optimal library of DMPs for generating the fundamental MP trajectories in PT, which is also generalizable to other tasks(see Section III-E).

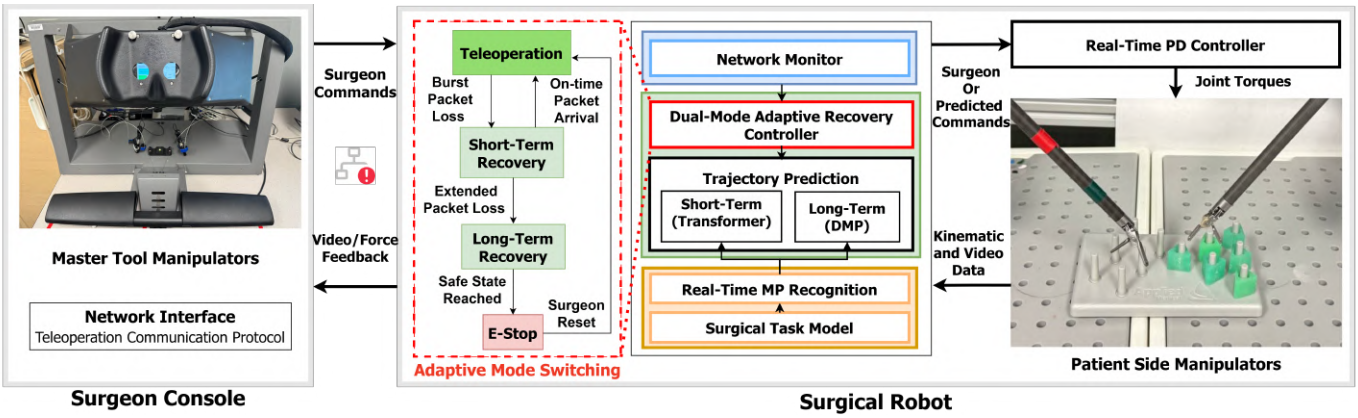

Fig. 2: Overall architecture of the dual-mode activity-aware recovery framework for teleoperated surgery. A surgeon controls a remote surgical robot using master tool manipulators. A dual-mode recovery controller intercepts surgeon commands on the remote robot side, and generates predicted recovery commands in the event of a communication interruption using dual-mode trajectory prediction models. An adaptive mode switching module mediates between teleoperation and recovery modes.

III. METHODS

In a teleoperated surgical system, a surgeon at a local console controls a remote, patient-side robotic platform. As shown in Figure 2, the surgeon's hand movements are captured by master tool manipulators and translated into digital commands. These commands are transmitted over a communication network, such as 5G, to the remote surgical robot. To create an immersive sense of telepresence, a feedback loop sends high-definition 3D video and haptic force data from the surgical environment back to the surgeon console, restoring the crucial senses of sight and touch.

We propose an *activity-aware* recovery framework that operates on the patient-side robot to autonomously handle *packet drop* and *communication loss*. Our framework intercepts surgeon commands to monitor network health and, when necessary, generates predictive commands based on the knowledge of current surgical task, to ensure safe and continuous operation. The architecture consists of a *Network Monitor*, a *Dual-Mode Adaptive Recovery Controller*, a *Real-time Movement Primitive Recognition* model, and activity-aware *Trajectory Prediction* models, as described next.

A. Network Monitor

The framework on the patient-side robot intercepts surgeon command packets, denoted by u_s, where $s \in \mathbb{N}$ is the packet sequence number. Each packet contains the desired incremental changes to the surgical instruments' state, including position ($\mathbf{p} \in \mathbb{R}^3$), quaternion orientation ($\mathbf{q} \in \mathbb{R}^4$), and gripper angle ($g \in \mathbb{R}$), and is represented as $u_s = (\Delta\mathbf{p}, \Delta\mathbf{q}, \Delta g)$. These packets are transmitted from the surgeon's console over a low-latency UDP-based protocol [37].

The Network Monitor receives these command packets and records the reception time for each packet u_s as $t_{p,s}$. The packet data is then decoded and formatted into a standard command for the recovery controller. The monitor continuously tracks the sequence number of the last received packet, s_{last}, and its corresponding reception time, $t_{p,s_{last}}$ to assess network health. This is done by analyzing packet inter-arrival times within an observation window of the w_{obs}^{net} most recent packets. At each control iteration $k \in \mathbb{N}$, the set of inter-arrival times, $\mathcal{I}(k)$, are queried by their corresponding reception times $\{t_{p,s_i}\}_{i=1}^{w_{obs}}$:

$$\mathcal{I}(k) = \{t_{p,s_{i+1}} - t_{p,s_i} \mid i \in \{1, \ldots, w_{obs}^{net} - 1\}\}$$

The Network Monitor calculates the median $\hat{M}(k)$ and standard deviation $\hat{\sigma}(k)$ of the values in $\mathcal{I}(k)$ and provides these statistics along with s_{last} and $t_{p,s_{last}}$ to the recovery controller.

B. Dual-Mode Adaptive Recovery Controller

The recovery controller is the central decision-making unit that uses data from the Network Monitor to orchestrate the system operation and recovery strategy. At each control-loop iteration k, executed at time t_k, the recovery controller determines the current operational mode of the system and generates the control command $\tilde{u}_k$ to be executed on the robot. This command is either directly from the surgeon ($\tilde{u}_k = u_s$) under normal teleoperation, or generated by the trajectory prediction models when one of the recovery modes is active. The commands are accumulated if multiple in-order packets are received within the same control iteration, resulting in a single consolidated motion command. These commands are then converted into a trajectory, which is tracked by a low-level Proportional-Derivative (PD) controller. The PD controller computes desired joint states via the robot's inverse kinematics, and outputs the corresponding motor torques. The details of our dual-mode adaptive recovery are presented next.

1) Recovery Modes: As shown in the state machine in Figure 2, the controller operates in four modes. The default *Teleoperation* mode directly forwards surgeon commands to the robot under normal network conditions. When a burst packet loss is detected, the system switches to *Short-Term Recovery*, activating a Transformer-based prediction model. If the disruption persists, it transitions to *Long-Term Recovery*, where a DMP-based model autonomously completes the surgical task. Once in long-term recovery, upon reaching a safe state (e.g., completing an object transfer to a target pole in the PT task), the system enters *E-Stop* mode, awaiting a surgeon-initiated reset to resume teleoperation.

2) Mode Switching Logic: The controller's switching logic executes at each robot control iteration k, as detailed in Figure 3. The time since the last received surgeon command is denoted by $\Delta t_k = t_k - t_{p,s_{last}}$. If a new packet with the expected sequence number ($s = s_{last} + 1$) arrives before a dynamic timer $T_{\text{switch}}(k)$ expires ($\Delta t_k < T_{switch}(k)$), it is processed normally. If an out-of-order packet ($s > s_{last} + 1$) is received, an $Out\_of\_Order()$ signal is raised, taking the system temporarily to the short-term recovery state, where the short-term recovery model generates the missing ($s - s_{\text{last}} - 1$) commands before forwarding the received one. Another transition from the teleoperation state occurs when no new packet arrives before $T_{\text{switch}}(k)$ expires. The controller then enters the short-term recovery, generating predictions at intervals of $T_{\text{expected}}(k)$ for a maximum duration of T_{max}. Here, $T_{\text{expected}}(k)$ is the expected time for new packet arrivals. Failure to re-establish communication within T_{max} leads to switching to long-term recovery, where an automated trajectory generator drives the system to a safe state, represented by the current transfer being complete ($Transfer\_Complete()$). Both $T_{\text{switch}}(k)$ and $T_{\text{expected}}(k)$ are dynamically updated at each control iteration. The set $\mathcal{I}(k)$ is adjusted after an out-of-order or packet loss event to account for missing or predicted sequence numbers.

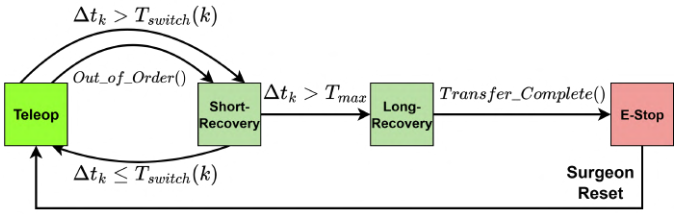

Fig. 3: Recovery mode switching of the dual-mode adaptive recovery controller. $\Delta t_k = t_k - t_{p,s_{last}}$ is the time since the last received packet and $T_{\text{switch}}(k)$ is a dynamic timer. The operating mode of the system transitions between manual teleoperation under normal conditions, short-term recovery when a burst loss happens, and long-term recovery once a prolonged communication loss occurs.

3) Adaptive Mode Switching: The performance of switching from teleoperation to short-term and long-term recovery modes heavily depends on the switching timer $T_{\text{switch}}(k)$, expected arrival timer $T_{\text{expected}}(k)$ and the maximum loss tolerance T_{max}. Our goal is to detect packet loss quickly and precisely to provide accurate compensation, while minimizing interference with the surgeon's operation.

One possibility is to statically estimate $T_{\text{expected}}(k)$ using either the median or mean of recent inter-arrival times over an observation window. However, as shown in 4 (a), using a median estimate ($\hat{M}(k)$) can lead to premature detection of false positive losses (which could be mitigated by adding a dynamic margin $\hat{\sigma}(k)$), while a mean-based timer ($\hat{\mathbb{E}}(k)$) shown in (b) may not detect actual packet losses, due to long-tailed exponentially distributed delays common in network traffic.

We propose an adaptive switch timer that takes into account both the median and the variations in recent packet arrival times, as well as commanded surgeon movements to do

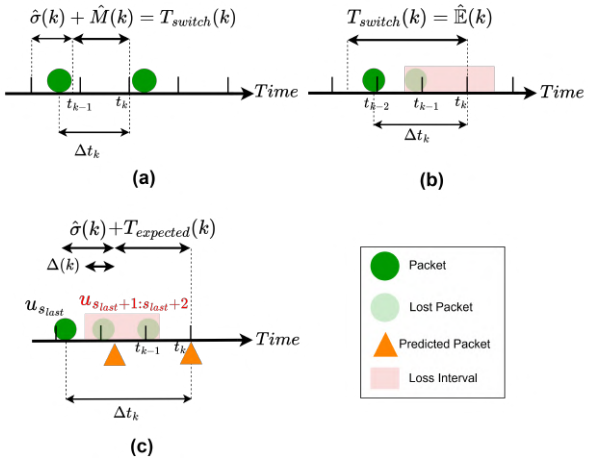

Fig. 4: Example packet transmission and recovery switching in lossy network conditions when using (a) median and (b) mean inter-arrival time estimates, vs. (c) our adaptive switching method. Packets received in correct time and order (Green) are sent directly to the robot, while lost packets are replaced by the predictively generated packets. The mean timer might miss losses, whereas using median timer without a margin can cause un-needed recovery switches. An adaptive margin can lead to quicker detections, especially for larger commands.

accurate and preemptive recovery actions. Specifically, we choose $T_{\text{expected}}(k)$ to be the median of inter-arrival times, $\hat{M}(k)$. The switching timer is then expressed as:

$$T_{\text{switch}}(k) = \hat{M}(k) + \Delta(k)$$

where $\Delta(k)$ is a dynamic margin:

$$\Delta(k) = c_1 \cdot c_2(\tilde{u}_{k-1}) \cdot \hat{\sigma}(k)$$

$$c_2(\tilde{u}_{k-1}) = \gamma + (1-\gamma) \cdot \tanh[\delta\,(1 - d_{k-1})]$$

with d_{k-1} as the largest normalized position, orientation or gripper movement command from the previously executed robot command:

$$d_{k-1} = \max_{i \in \{1,\ldots,8\}} \left(\frac{|\tilde{u}^i_{k-1}|}{|\tilde{u}^i|_{max}} \right) \tag{1}$$

and the normalization maximum $|\tilde{u}^i|_{max}$, which is the value of maximum command along axis i across all MPs, obtained from a train set. The constants c_1, $0 < \gamma < 1$ and δ are tuned adaptively using recent packet data to minimize packet loss detection errors.

The inclusion of $\gamma \leq c_2(k) \leq 1$ allows the controller to adjust its sensitivity to packet loss based on the magnitude of the last executed command; loss of larger motions may lead to greater errors and thus require quicker recovery. For any value of γ, $c_2(k)$ is a decreasing function of d_k, meaning larger movements make the margin $\Delta(k)$ smaller and the packet loss timer triggers more quickly.

Figure 4 (c) demonstrates a period of burst packet loss ($k-2 : k-1$) that occurs before the current iteration k. The standard deviation margin $\hat{\sigma}(k)$ is shown against our adaptive margin $\Delta(k)$. The $\hat{\sigma}(k)$ margin alone could potentially lead

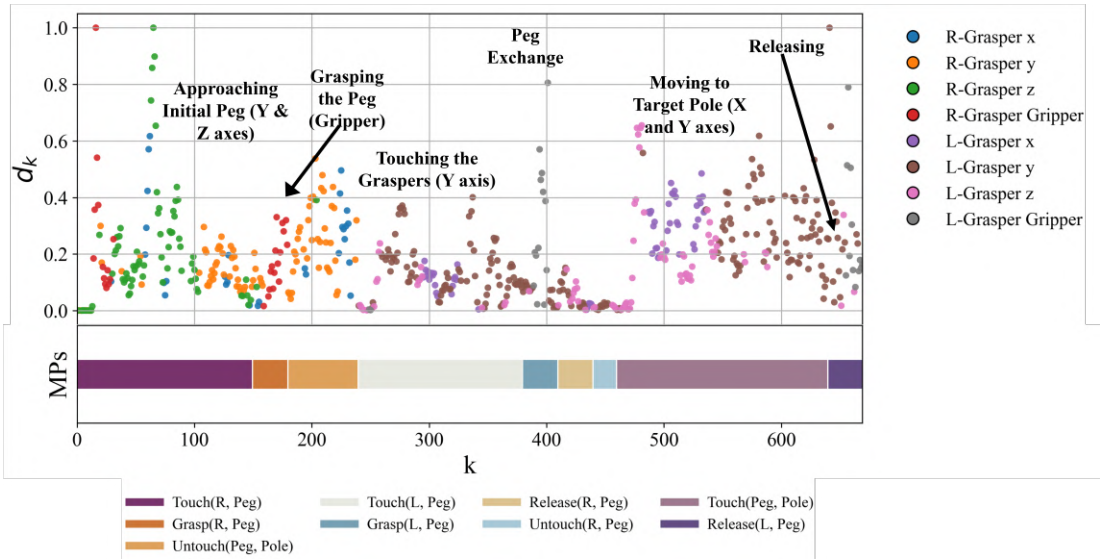

Fig. 5: Demonstration of the largest normalized commands (d_k) at each control iteration k (top), and action segments (MPs) during one peg transfer (bottom). Each colored point represent the axis normalized value of the largest command at each iteration. During each MP, large movement commands are observed along certain axes, which are unique to and critical for that MP. Data has been downsampled to 30Hz for better visualization.

to late or missed detection of losses, which can especially be undesirable during large movements (e.g. if $u_{s_{last}+1:s_{last}+2}$ carries large movement commands). Here, for instance, this margin is failing to detect the burst loss whereas the adaptive margin can compensate for it by shrinking the $T_{switch}(k)$ when needed, enabling proactive or more timely intervention.

Figure 5 shows the MPs for one PT trajectory along with the largest normalized command (d_k in eq. 1) for each timestep in the trajectory. At certain points during each MP, the value of one or more kinematic commands becomes relatively large (e.g., Z-axis position when approaching and Gripper angle when grasping or releasing the peg). These are times when a loss of commands could lead to large deviations from surgeon's intent and potential errors like a peg drop, as large commands at one timestep are often accompanied with large commands in the following steps. In these situations, our adaptive timer enables preemptive recovery actions by switching to our autonomous trajectory generation to prevent potentially large errors (see V-B).

C. Real-time Movement Primitive Recognition

For activity-aware recovery, we incorporate the knowledge of the current task and MP into our recovery trajectory prediction. Specifically, we use a custom multi-modal Transformer model based on [38], trained on task-specific data, to recognize the currently executed MP in real-time. Unlike previous filtering-based methods [14], [15], this approach enables extracting complex feature relationships and higher-level abstractions to closely align with the surgeon's intentions.

For real-time MP recognition, we incorporate a feedback loop of kinematic and video data. The kinematic inputs (K_{11}) include the instruments' position ($\mathbf{p}$), orientation ($\mathbf{q}$), gripper angle (g), and velocity ($\mathbf{v} \in \mathbb{R}^3$), all derived from joint encoders via forward kinematics. The model predicts the current MP based on the K_{11} features ($K_{11}[-w_{obs}^{kin}:]$) and Red-Green-Blue (RGB) frames of surgical scene video feedback (encoded individually using an ImageNet pretrained ResNet [39] model) for an observation window of length w_{obs}^{kin} and outputs the predicted MP to the short-term and long-term trajectory prediction models.

D. Short-Term Recovery from Packet Loss

For accurate short-term recovery, we use the same Transformer model based on [38] for trajectory prediction. The model predicts a future trajectory window of length w_{pred}^{kin} using the K_{11} features ($K_{11}[-w_{obs}^{kin}:]$), RGB frames (encoded using a ResNet model) and recognized MP labels from an observation window of length w_{obs}^{kin}. With access to the MP labels, automatically generated in the recognition stage of the pipeline, the transformer model makes more informed predictions by incorporating contextual information from the task dynamics. This model runs continuously in the background, providing immediate predictions during packet loss. While this approach has demonstrated good accuracy on dexterous surgical tasks [38], [40], large prediction windows can lead to error accumulation. To address this, we introduce random-walk noise from a Gaussian distribution $\mathcal{N}(0, \sigma_v = 0.0003)$ to the K_{11} features during training following [41], which helps the model handle the errors encountered in rollouts. Additionally, since inconsistencies demonstrated by the novice subjects during the execution of surgical actions can deteriorate the performance of learning based methods [38], we apply curriculum learning [42] by progressively training on demonstrations from subjects with increasing expertise, starting with novices and advancing to intermediates and experts.

E. Long-Term Recovery from Communication Loss

In long-term recovery, the system predicts a sequence of MPs to move the robot to the closest safe state, using the

current recognized MP and pose of the robot. This MP recovery sequence is generated based on the current MP recognized by the transformer model, and the sequence of next MPs required to finish the current transfer of the peg. Based on the *surgical task model*, represented as a grammar graph consisting of actions (e.g. MPs) and their semantic targets (see Figure 1), a series of MPs are generated by the long-term recovery module to take the system to a safe state. For the purpose of our study, a *safe state* is defined as a completed transfer, where all pegs are stationary placed on a unique pole, and instruments are all in the field of view with no collisions.

To achieve stable long-term trajectory generation, we build a library of DMPs corresponding to the MPs in the task grammar graph, and, at the time of recovery, generate the predicted trajectories for each MP in the sequence. These piece-wise trajectories are then "stitched" together back-to-back to build the recovery trajectory consisting of one or more MPs, until a safe state is reached (i.e., the transfer of the current peg is completed).

To generate a trajectory using a DMP at runtime, a start pose and a target or goal pose are required. The start pose of each MP is the target pose of the previous MP in the recovery sequence. The target poses for execution of an MP sequence can be inferred from calibrated 3D visual inputs like RGB+Depth cameras. Although certain targets like peg exchange and release poses can be calculated optimally based on known board and environment geometry, the initial grasp point requires a pipeline consisting of object detection, pose estimation and grasp point generation, similar to [29], [25].

For learning DMPs for each MP, we study the following state-of-the-art DMP models for both position and orientation trajectories and select the DMP model that achieves lowest average error in generating the MP's position trajectories.

1) Cartesian Movement Primitives: In Cartesian space, DMPs can encode robot position $\mathbf{p}$ and orientation $\mathbf{q}$ in 3D space. The DMP equations for position are:

$$\tau\dot{\mathbf{v}} = \alpha_p(\beta_p(\mathbf{p}_g - \mathbf{p}) - \mathbf{v}) + \mathbf{f}_p(\theta), \quad \tau\dot{\mathbf{p}} = \mathbf{v} \tag{2}$$

And for orientation:

$$\tau\dot{\boldsymbol{\eta}} = \alpha_q\left(\beta_q \log(\mathbf{q}_g * \mathbf{q}^{-1}) - \boldsymbol{\eta}\right) + \mathbf{f}_q(\theta), \quad \tau\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\eta} * \mathbf{q} \tag{3}$$

where τ is a temporal scaling factor, α_p, α_q β_p and β_q are gain parameters, $\mathbf{p}_g$ and $\mathbf{q}_g$ are the target position and orientation, and $\mathbf{f}_p(\theta)$, $\mathbf{f}_q(\theta)$ are nonlinear forcing terms for position and orientation respectively. The canonical system governs the evolution of the phase variable θ:

$$\tau\dot{\theta} = -\alpha_\theta\theta \tag{4}$$

The nonlinear forcing term $\mathbf{f}(\theta)$ is defined as:

$$\mathbf{f}(\theta) = \frac{\sum_{i=1}^{N}\psi_i(\theta)\mathbf{w}_i}{\sum_{i=1}^{N}\psi_i(\theta)}\theta \tag{5}$$

where $\psi_i(\theta) = \exp(-h_i(\theta - c_i)^2)$ are Gaussian basis functions with centers c_i, widths h_i, and weights $\mathbf{w}_i$. During training, the weights $\mathbf{w}_i$ are learned such that the generated trajectory closely matches the shape and characteristics of expert demonstrations. We use the Cartesian DMP formulation from [43].

2) Probabilistic Movement Primitives: Probabilistic Movement Primitives (ProMPs) [44] extend DMPs by introducing a probabilistic model for handling variability across multiple demonstrations. A trajectory is modeled as:

$$\mathbf{p}(t_k) = \boldsymbol{\Phi}(t_k)^\top\mathbf{w} + \boldsymbol{\epsilon} \tag{6}$$

where $\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)$ are the learnable parameters, $\boldsymbol{\Phi}$ represents a vector of Gaussian basis functions, and ϵ represents a random noise. ProMPs generate new trajectories by sampling from this distribution, ensuring generalization while preserving statistical properties.

3) Conditional Neural Movement Primitives: Conditional Neural Movement Primitives (CNMPs) [45] generate movements conditioned on external goals (target poses) and sensorimotor (kinematic) inputs. CNMPs predict a distribution over the target kinematic values using an encoder and query network. This allows the model to adapt trajectories based on real-time sensory feedback (including MP transitions), ensuring flexibility and robustness in dynamic environments. We use the DMP target poses to condition the generated trajectory.

4) Learning from Successful and Failed Demonstrations (TLFSD): This method [46] optimizes trajectory reproduction by learning from both successful and failed demonstrations. The successful and failed sets are encoded as Gaussian Mixture Models with corresponding mean trajectories. It utilizes a cost function that penalizes deviations from successful demonstrations and rewards divergence from failed ones. We use the success and failure labels assigned to the execution of gestures in our datasets as labels for training TLFSD.

IV. Experimenteal Setup

We conduct a comprehensive set of experiments to evaluate the effectiveness of each component as well as the end-to-end dual-mode recovery framework in mitigating errors. We first assess the performance of the short-term recovery model in handling transient packet loss (Section V-A), followed by an analysis of the adaptive mode-switching logic (Section V-B), and the evaluation of DMP-based long-term recovery (Section V-C). Finally, we test the integrated end-to-end framework to demonstrate the synergistic benefit of our dual-mode approach in ensuring operational safety and continuity during severe network disruptions (Section V-D).

A. Datasets

Real-time teleoperation sessions are simulated using data collected from two surgical teleoperation setups with realistic network emulation. The Leave-One-Super-trial-Out (LOSO [47]) split is used for cross-validation. At each iteration of LOSO, we leave one trial from each subject out as test set, train the model on the remaining trials and compute the average performance across iterations.

We use the following datasets for our evaluation:

DESK [19], a publicly-available dataset consisting of kinematic and video data and gesture labels from 48 trials of PT performed on a da Vinci surgical robot by 8 subjects (2 novices, 4 intermediates, and 2 experts), each performing 6

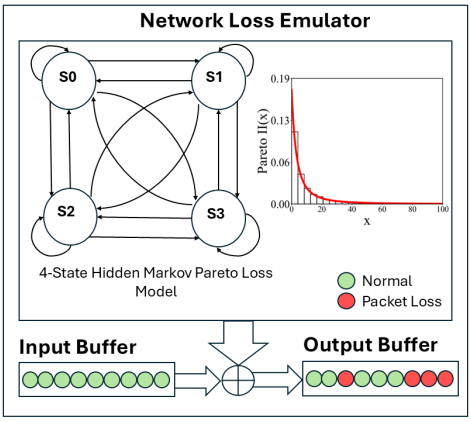

Fig. 6: Packet loss emulation pipeline and model. The packets are lost through a stochastic process, where the the length of packet loss is determined by a Pareto type II (lomax) model, and the parameter of the model at each instance (and hence the QoS of the network, by proxy) are determined by a four-state Markov model. Communication loss in the context of teleoperation is modeled as a burst loss of at least two seconds.

trials of 3 peg transfers. We train the model on 40 trials and test on 8 left-out trials (256 test MPs on average). Synchronized MP labels are obtained from the COMPASS dataset [23].

RAVEN, which is a collection of 15 trials of PT performed on an open-source Raven surgical robot [20] by two engineering researchers with moderate skill levels. We annotated this dataset for MPs using recorded RGB videos and an open-source annotation software [48]. We only evaluate the end-to-end performance of our system using this dataset.

B. Packet and Communication Loss Emulation

Packet loss in wireless networks is often bursty, degrading performance in real-time applications like telesurgery. To model this, we use a four-state Gilbert-Elliott (GE) Markov chain model [49] with a heavy-tailed Burst Loss Length (BLL) distribution [50]. This model captures temporal correlation of losses in WLANs, common in hospital networks with interference and contention, and has been validated for LTE/4G networks with parameter tuning [51].

The Markov chain includes four states: S_0 (Good), S_1 (Bad), and intermediate states S_2, S_3, representing different packet loss levels. Transitions between states follow probabilities $P_{ij} = \mathbb{P}(S_{k+1} = S_j \mid S_k = S_i)$, where $\sum_{j=0}^{3} P_{ij} = 1$ for all $i \in \{0, 1, 2, 3\}$.

The BLL X in state i is modeled using a Pareto Type II (Lomax) distribution with the probability density function:

$$f_i(x) = \frac{\nu_i \lambda_i^{\nu_i}}{(x + \lambda_i)^{\nu_i + 1}}, \quad x > 0, \quad \nu_i, \lambda_i > 0 \tag{7}$$

with ν_i controlling tail heaviness and λ_i the scale. Samples are drawn via inverse transform sampling, where $\mathcal{U}$ represents a uniform distribution:

$$X = \lambda_i \left((1 - U)^{-\frac{1}{\nu_i}} - 1 \right), \quad U \sim \mathcal{U}(0, 1) \tag{8}$$

ensuring burst lengths follow the desired heavy-tailed behavior. Figure 6 shows the emulation pipeline. The state transitions are adopted from [50], and values of Lomax parameters (ν_i and λ_i) are obtained through a Monte-Carlo simulation and differential evolution optimization for targeted packet loss rates in experiments (5% to 50%). Communication loss is emulated as a complete interruption in packet transmission for at least two seconds.

C. Baselines

1) Mode Switching Baselines: To benchmark our adaptive switching mechanism, we compare it against three intuitive, non-context-aware statistical baselines (see SubsectionIII-B3. The Mean timer $T_{switch}(k) = \hat{E}(k)$ uses the average inter-arrival time as a simple expectation for packet arrival. We also test a Median timer $T_{switch}(k) = \hat{M}(k)$, which offers a more robust estimate of central tendency against the skewed, long-tailed delays common in network traffic. Finally, a Median+Standard Deviation timer $T_{switch}(k) = \hat{M}(k) + \hat{\sigma}(k)$ provides a more conservative baseline by adding a statistical margin to account for network jitter. T_{max} is empirically set to two seconds, which is sufficient to leverage the transformer model's high accuracy for shorter-term horizons and more intricate MPs, while allowing longer MPs to be covered by long-term recovery to prevent large rollout errors.

2) Trajectory Prediction Baseline: As a state-of-the-art baseline to evaluate against our proposed framework, an enhanced version of the Kalman filtering proposed by [15] is utilized, by incorporating orientation predictions via an Unscented Kalman Filter (UKF) [52] to account for the nonlinear orientation representation. Once packet loss or communication loss is detected, the UKF is applied using the last known position, orientation, velocity and acceleration of the instrument end-effectors, and new observations are simulated using the dynamic model of the robot, for both DESK (collected using da Vinci Research Kit [53]) and our Raven [20] robot.

D. Model Parameter Settings

1) Transformer-based Model: The transformer-based model adopted from [38] is trained using the same hyper-parameters as the authors suggested. We take both w_{obs}^{kin} and w_{pred}^{kin} to be 1 second, beyond which roll-out on the past predictions of model are used. The kinematic features and video frames ($K_{11} + RGB$) are used as input.

2) DMPs: Prior to training DMPs on the train set, the demonstrations of each MP is resampled to the median length of that MP using either down-sampling or linear interpolation for better consistency. Generation of a new trajectory involves step-wise integration until sufficient convergence to target is achieved. The poles and the other robotic arm are modeled as obstacles, and used as repulsive forces during trajectory generation [43]. For the Cartesian DMPs, we use 100 basis Gaussians with c_i constants spread uniformly across the median duration of each MP. For ProMPs, 7 Gaussian mixtures are used for each arm; one Gaussian per each axis. The values of α and β for both position and orientation are set such that $\alpha\beta = 3000$ and the differential equation is stable and converging, and we chose $\alpha_\theta = 4$.

E. Metrics

1) Trajectory Prediction: We use Root Mean Square Error (RMSE) to quantify the deviation of the predicted trajectories by the recovery framework from the ground truth trajectories, each represented as sequences of poses with position vectors $\mathbf{p}_i$ and quaternion orientations $\mathbf{q}_i$. Given an original trajectory $\{\mathbf{p}_i^{\text{orig}}, \mathbf{q}_i^{\text{orig}}\}$ and a predicted trajectory $\{\mathbf{p}_i^{\text{pred}}, \mathbf{q}_i^{\text{pred}}\}$, the position RMSE is:

$$\text{RMSE}_{\text{pos}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left\|\mathbf{p}_i^{\text{pred}} - \mathbf{p}_i^{\text{orig}}\right\|^2}, \tag{9}$$

and the orientation RMSE is:

$$\text{RMSE}_{\text{orient}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(2\arccos\left(\left|\mathbf{q}_i^{\text{pred}} \cdot \mathbf{q}_i^{\text{orig}}\right|\right)\right)^2}. \tag{10}$$

As the length of the ground truth and predicted trajectories might be different, we use the dynamic time warping algorithm to match corresponding data points on the two trajectories.

2) Mode Switching: We evaluate the effectiveness of different switching timers using three metrics: detection performance, latency, and the effect on recovery performance. Detection performance is quantified using False Positive Rate (FPR), False Negative Rate (FNR) and F1 score. We define a true positive (TP) as a timer trigger that occurs after at least one packet has been lost, with no newer packets having arrived. Conversely, a false positive (FP) is a trigger that occurs when no packets were lost. A false negative (FN) is recorded at a control cycle where at least one packet has been lost since its expected arrival time ($t_{p,last} + T_{expected}(k)$), but the timer has not yet triggered. Latency is the time between the missed packet's expected arrival time to the timer trigger (detection time). To measure the impact of timer triggers on the downstream recovery performance, once a timer is triggered, we generate the predicted commands and measure the resulting (predicted) trajectory's deviation from the ground-truth (intended) trajectory using RMSE.

V. Results and Discussion

A. Short-Term Recovery

To evaluate the effect of packet loss on robot instrument deviations from intended surgeon commands, we simulate scenarios with different packet loss rates and compute the resulting trajectory deviations across different MPs. Figure 7 shows trajectory deviations due to four packet loss severity levels. It is observed that dropping packets drives the end-effector off its intended path by up to nearly 26 mm at 50% loss, with the error rising monotonically with loss rate across all six MPs. The effects of packet loss are more severe during longer motions that involve faster maneuvers like $Touch$ and $Untouch$ actions. Adding a Kalman filter reduces these deviations, but only modestly (25–35% on average). Because the filter is blind to underlying context of surgical actions and prior human demonstrations, it cannot anticipate the changes in direction (e.g. raising and lowering the pegs) that dominate $Touch$ and $Untouch$ motions, leaving residual spikes of 6–11.7 mm. The proposed context-aware transformer

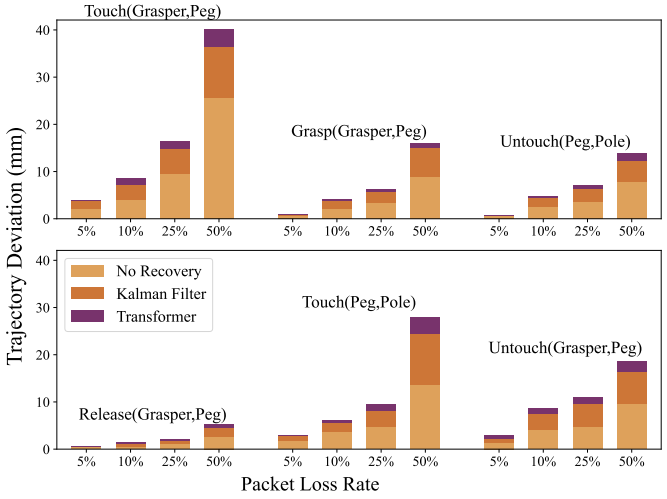

Fig. 7: Comparison of trajectory deviation due to four levels of packet loss from 5% to 50%, with and without two predictive models, a Kalman filter and our short-term recovery transformer. Predictive methods mitigate deviation for all MPs, with the transformer being the most accurate.

remedies this. Using action-specific dynamics, it reduces the worst-case error to < 4.5 mm and holds < 2 mm for loss rates $\leq 25\%$. The difference in effectiveness is most pronounced during higher loss rates, where a simple filtering method accumulates large errors over longer loss bursts, whereas a context-aware model is capable of compensating for the loss of information by incorporating an understanding of task context and generating predictions that are more aligned with human intentions. The model is capable of achieving real-time performance, as the latency trajectory generation remains bellow 1ms, which is within PD controller's 1 khz frequency.

TABLE II: Switching Timer Performance Evaluation. Packet Loss Detection False Positive Rate (FPR), False Negative Rate (FNR), F1 Score, Detection Latency (ms) and Resulting Trajectory Deviation (mm) are Presented.

Timer Type	FPR	FNR	F1	Latency (ms)	RMSE (mm)
Mean	**0.000**	0.33	**0.88**	131	12.34
Median	0.270	**0.02**	0.79	**85**	8.09
Median+Std Dev	0.002	0.26	0.75	115	9.21
Adaptive	0.008	0.19	0.83	**94**	**3.62**

B. Recovery Mode Switching

Effectiveness of the proposed adaptive mechanism for switching from Teleoperation to short-term recovery is evaluated by simulating severe packet loss with a 25% loss rate and 50ms added exponential delay, which is typical in wireless networks. We use the same short-term recovery model as the previous section to fill the gaps caused by lost packets during simulation LOSO test sets. In Figure 8, the inter-arrival times are shown at the top, with lost packets highlighted by red dots, the middle plot shows the actual value of each timer at each timestep, and the bottom plot shows the burst loss arrivals and the triggering of each timer. The mean timer is seen to be too relaxed, while the median timer is too conservative, causing

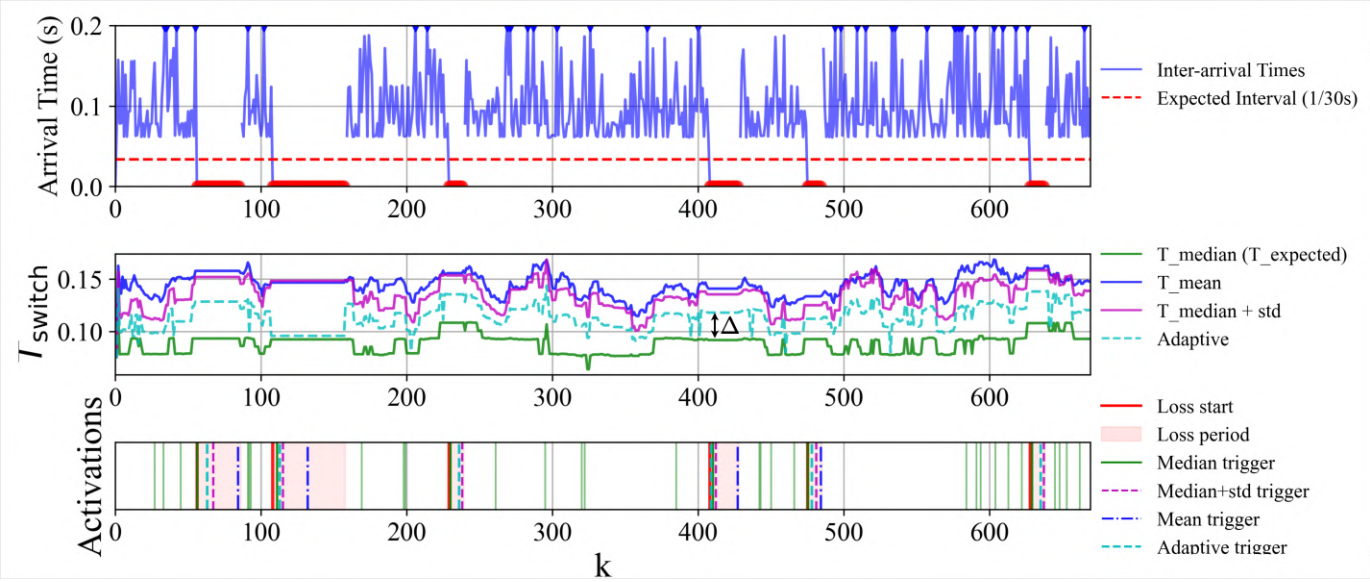

Fig. 8: Packet inter-arrival times in the presence of bursts of packet loss with average 25% loss rate and 50ms average delay. The value of different timers and their triggering are shown.

TABLE III: Comparison of Various DMP Models for Long-Term Recovery: Average RMSE for Trajectory Position (Pos) and Orientation (Ori) are reported in millimeters (mm) and radians, respectively.

Model	Touch(Grasper, Peg)		Grasp(Grasper, Peg)		Untouch(Peg, Pole)		Release(Grasper, Peg)		Touch(Peg, Pole)		Untouch(Grasper, Peg)	
	Pos	Ori	Pos	Ori	Pos	Ori	Pos	Ori	Pos	Ori	Pos	Ori
CNMP	2.11	0.34	**0.09**	0.018	1.89	0.07	**0.14**	0.006	2.65	0.32	**0.87**	0.031
CartesianMP	**1.67**	**0.11**	0.17	**0.009**	1.5	**0.03**	0.29	**0.013**	**1.38**	**0.1**	1.3	0.025
ProMP	1.84	0.2	0.16	0.013	1.46	**0.03**	0.22	0.016	1.77	0.24	1.09	0.03
TLFSD	2.09	0.27	0.13	0.031	**1.15**	0.04	0.15	0.014	1.95	0.27	0.92	0.036
Kalman Filter	5.92	0.021	1.3	0.14	2.23	0.07	1.27	0.061	5.02	0.35	4.42	0.04
Transformer	3.83	0.17	0.95	0.12	1.51	0.08	0.51	0.041	3.34	0.33	3.47	0.05

unnecessary switches. The adaptive timer acts as a middle ground, accounting for potential hazards resulting from the loss of large movement commands.

The results for various timer types are presented in Table II. Median timer has the lowest detection latency and has higher recall (smaller false negative rate) than mean timer, but is less precise (higher false positive rate) and also can't sufficiently compensate for trajectory deviation. The mean timer is too relaxed as it has a very low false positive rate, but it also misses many of actual packet loss bursts and is the least effective approach with the largest delay and deviation. Augmenting median timer with information about the inter-arrival time variations (median+std) also has major detection issues in capturing actual losses, but is very precise. The adaptive timer, although not the most effective in capturing all losses, provides the best tradeoff between latency and performance and results in the least deviation by preventing major deviations resulting from losses of large-movement commands.

C. Long-Term Recovery

Long-term recovery generates trajectories for an entire MP, from an starting pose to a target pose. Thus, various DMPs are tested by training them on the training set and testing them by generating entire MP trajectories for MPs from the test set. The results of evaluating the four DMP models for each MP are presented in Table III. Notably, for predicting longer trajectory segments that encompass an entire MP, DMPs consistently demonstrate similar or superior performance compared to the Transformer and Kalman Filter models. In MPs requiring intricate wrist maneuvers for orientation adjustments, such as $Touch(Grasper, Peg)$ and $Touch(Peg, Pole)$, Cartesian DMP clearly excels. For the short-duration MP $Release$, the performance of the Transformer and DMPs is comparable. However, for longer MPs like $Touch$ and $Untouch$, the difference in performance becomes more pronounced, with DMPs outperforming the alternatives.

In the MP with the highest number of unsuccessful demonstrations, $Untouch(Peg, Pole)$, the TLFSD method significantly outperforms the others. It is empirically observed that this method learns trajectories that would lift the peg vertically with minimal excessive lateral motion while keeping the orientation consistent. Furthermore, in most MPs, CNMP closely matches the best performing DMP, and it stands out as the best choice for the $Untouch(Grasper, Peg)$ MP. Based on these evaluations, we construct the long-term recovery framework by selecting the most suitable DMP for each MP: Cartesian DMP for predicting $Touch(Grasper, Peg)$ and $Touch(Peg, Pole)$, CNMP for $Grasp(Grasper, Peg)$, $Release(Grasper, Peg)$, and $Untouch(Grasper, Peg)$, and TLFSD for $Untouch(Peg, Pole)$. This strategy ensures op-

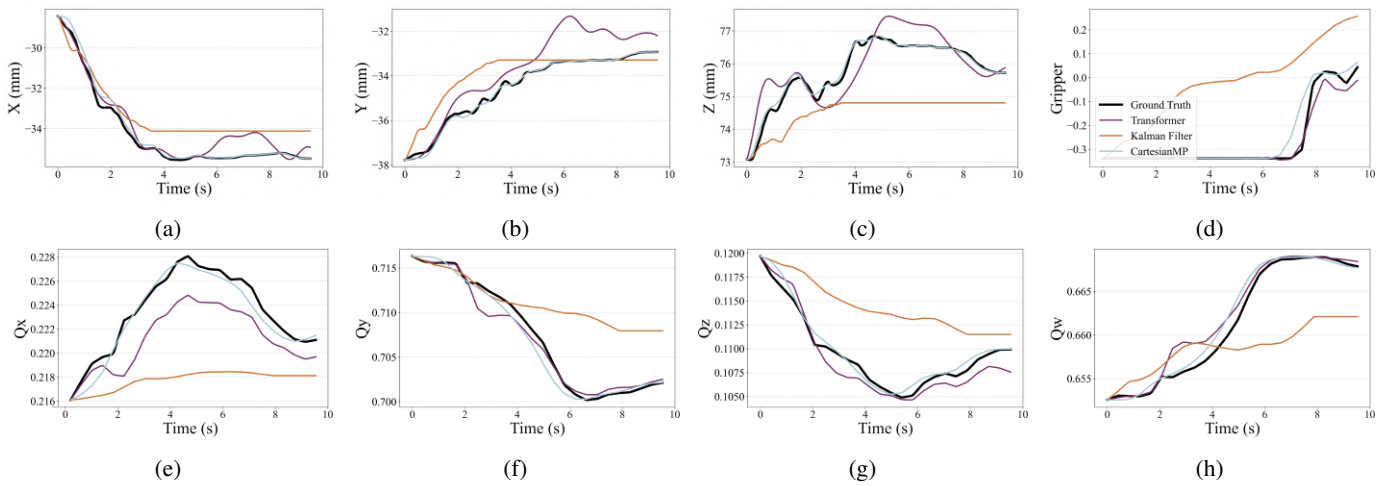

Fig. 9: Comparison of CNMP, Transformer, and Kalman Filter for long-term trajectory prediction of a $Grasp(Grasper, Peg)$.

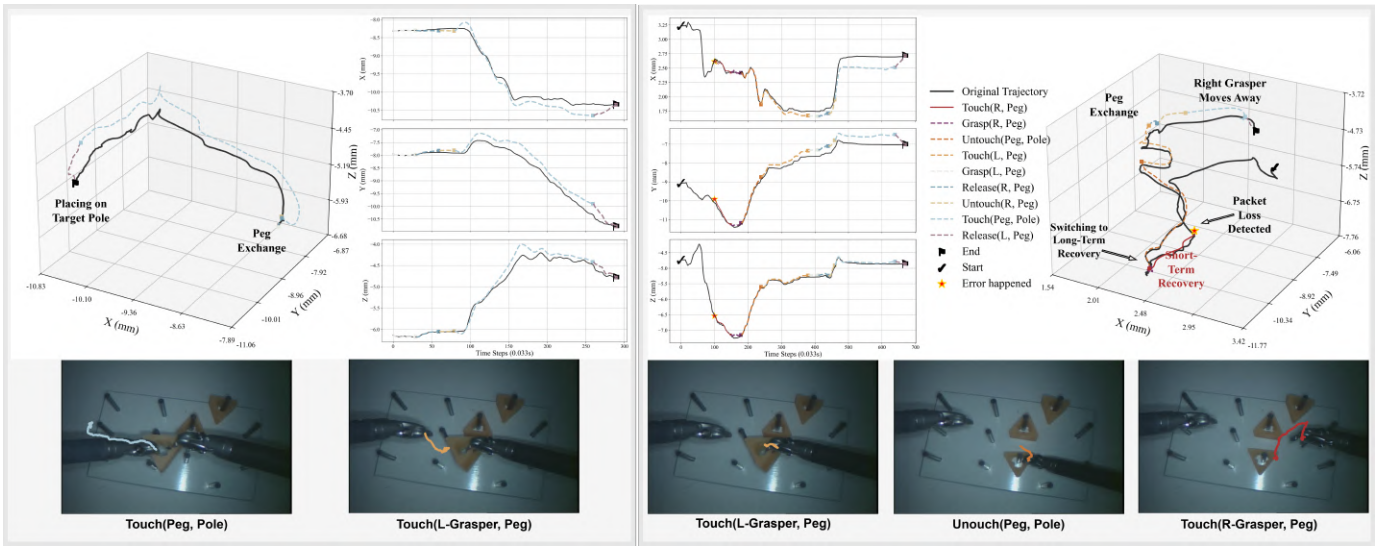

Fig. 10: End-to-end demonstration of the recovery framework, under a short period of 50% packet loss, followed by complete loss of communication. The scenario is simulated on one of the test trials from DESK dataset. The beginning of error falls withing the first Touch movement primitive to the target object. **Right:** the trajectory of the right grasper is depicted in 3D, and along three Cartesian axes and the mapped trajectory onto RGB images. The predicted trajectory consists of short-term transformer trajectory (solid red segment), and continued by long-term DMP trajectories (dashed segments). **Left:** The predicted trajectory of the left grasper, from the peg exchange until the transfer is complete, generated entirely by DMPs.

timal performance for optimal recovery trajectory generation. Figure 9 illustrates an example of the trajectories generated by CNMP, Kalman filter, and transformer model. We observe that the CNMP model, which was the top DMP representation for the $Grasp(Grasper, Peg)$ MP, invariably outperforms the other methods by a large margin, and is more effective in capturing the long-term characteristics of this MP.

D. End-to-End Recovery

To evaluate the end-to-end performance of our recovery framework, we simulate communication interruptions at different times during the task execution. Specifically, we initialize trials from the test set at random points uniformly along each MP and then simulate a complete communication failure (see Section IV-B). We then allow the recovery framework to take the system to a safe state, which occurs when the current transfer is completed and the final MP of the task is executed. We do this 30 times for each MP in the test set, and report the average deviation in the predicted trajectories. Figure 10 illustrates an example in which network interruption begins mid-way through approaching a peg and the $Touch(Grasper, Peg)$ MP, prompting the recovery framework to generate and execute a recovery trajectory. Short-term trajectory generation for the $Touch$ MP is followed by long-term trajectory generation for the sequence of following MPs until transfer is complete. The results of this experiment are summarized in Table IV, where the performance of the end-to-end framework is compared against using only the Transformer model or only the DMP models (in their best combination). The findings for both DESK dataset and our Raven dataset demonstrate that the end-to-end recovery framework outperforms the alternatives

TABLE IV: Ablation Evaluation of Transformer Model, DMP Models, and End-to-End Recovery Approach. Average RMSE for Trajectory Position (Pos) and Orientation (Ori) are reported in millimeters (mm) and radians, respectively, on both the DESK and Raven datasets.

Dataset	Method	Touch(Grasper, Peg)		Grasp(Grasper, Peg)		Untouch(Peg, Pole)		Release(Grasper, Peg)		Touch(Peg, Pole)		Untouch(Grasper, Peg)	
		Pos	Ori	Pos	Ori	Pos	Ori	Pos	Ori	Pos	Ori	Pos	Ori
DESK	**Transformer Only**	9.67	0.285	6.77	0.29	5.625	0.135	3.635	0.06	3.16	0.585	2.53	0.087
	DMP Only	5.075	0.195	3.75	0.30	3.23	0.075	2.255	0.0105	2.05	0.18	1.64	0.053
	End-to-End Recovery	**3.85**	**0.15**	**3.135**	**0.245**	**2.865**	**0.06**	**1.71**	**0.058**	**1.325**	**0.165**	**1.038**	**0.039**
Raven	**Transformer Only**	9.94	0.258	6.47	0.27	5.891	0.140	3.92	0.06	3.11	0.530	2.39	0.087
	DMP Only	4.594	0.183	3.86	0.30	**3.05**	0.076	2.395	**0.0095**	2.18	0.19	1.59	0.049
	End-to-End Recovery	**4.20**	**0.25**	**2.880**	**0.225**	3.064	**0.067**	**1.82**	0.061	**1.335**	**0.181**	**1.013**	**0.059**

in most cases by large margins, with the adaptive dual-mode recovery algorithm proving more effective than using either recovery method in isolation. Going from MPs often executed early in a transfer to those executed more towards the end of a transfer, it is evident that the deviation error generally decreases, and MPs with longer duration and motion lengths tend to be more affected by network communication losses. As the same models of pegboard and peg are used in both datasets, the error values are close and comparable, and the insights are similar, although in Raven the orientation compensation seems to be slightly sub-par compared to results from DESK.

VI. CONCLUSION

This paper introduces an activity-aware, dual-mode recovery framework to ensure safety and operational continuity in teleoperated surgery during severe network disruptions. Our primary contribution is a shared control architecture that intelligently switches between two recovery strategies: a Transformer-based model for short-term prediction and a library of specialized DMPs for long-term task completion.

Our experimental results validate the effectiveness of this approach. The proposed adaptive switching timer, which leverages motion-intent, achieved the most balanced performance in detecting packet loss and mitigating trajectory deviation. In short-term recovery, our Transformer model drastically reduced trajectory deviation compared to simpler filtering methods. For long-term communication loss, our hybrid DMP approach, which selects the optimal DMP model for each surgical primitive, proved superior for generating complete, accurate trajectories. Most importantly, the end-to-end framework outperformed using only short-term or long-term recovery modules in isolation, confirming the synergistic benefit of the dual-mode design. This performance was consistently demonstrated on datasets from two different surgical robot platforms, indicating the framework's generalizability.

Despite these promising results, this study has several limitations. First, the framework was evaluated on the Peg Transfer task, which, while a standard benchmark, has a linear task grammar graph and lacks the complexities of operating on deformable tissues. Second, all evaluations were conducted using simulations based on pre-recorded data, without validating the system's performance and the smoothness of transitions in real-time, human-in-the-loop settings. Future work will focus on addressing these limitations to advance the framework toward real-world clinical applications. We plan to extend our evaluation to more complex tasks, such as suturing, which involve multi-branch decision-making and interaction with soft tissues. Additionally, conducting human-in-the-loop studies with surgeons will be critical to assess the practical utility, safety, and user acceptance of the system.

REFERENCES

[1] S. K. Singh, J. Sharma, L. M. Joshua, F. Huda, N. Kumar, and S. Basu, "Telesurgery and robotics: Current status and future perspectives," in *Telehealth and Telemedicine - The Far-Reaching Medicine for Everyone and Everywhere*, T.-C. Wang, Ed. London: IntechOpen, 2022, ch. 6. [Online]. Available: https://doi.org/10.5772/intechopen.107465

[2] Intuitive Surgical, Inc., "Intuitive sustainability report," https://www.intuitive.com/en-us/-/media/ISI/Intuitive/Pdf/2024-Intuitive-ESG-Report.pdf, 2021, [Online; accessed 01-March-2023].

[3] V. P. Gurupur and Z. Miao, "A brief analysis of challenges in implementing telehealth in a rural setting," *mHealth*, vol. 8, p. 17, 2022, published April 20, 2022. [Online]. Available: https://doi.org/10.21037/mhealth-21-38

[4] P. Garcia, J. Rosen, C. Kapoor, M. Noakes, G. Elbert, M. Treat, T. Ganous, M. Hanson, J. Manak, C. Hasser *et al.*, "Trauma pod: a semi-automated telerobotic surgical system," *International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 5, no. 2, pp. 136–146, 2009.

[5] C. R. Doarn, M. Anvari, T. Low, and T. J. Broderick, "Evaluation of teleoperated surgical robots in an enclosed undersea environment," *Telemedicine and e-Health*, vol. 15, no. 4.

[6] D. Pantalone, G. S. Faini, F. Cialdai *et al.*, "Robot-assisted surgery in space: pros and cons. a review from the surgeon's point of view," *npj Microgravity*, vol. 7, p. 56, 2021.

[7] e. a. H. Alemzadeh, "Targeted attacks on teleoperated surgical robots: dynamic model-based detection and mitigation," in *46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2016, pp. 395–406.

[8] S. Xu, M. Perez, K. Yang, C. Perrenot, J. Felblinger, and J. Hubert, "Determination of the latency effects on surgical performance and the acceptable latency levels in telesurgery using the dv-trainer® simulator," *Surgical Endoscopy*, vol. 28, no. 9, pp. 2569–2576, Sep. 2014. [Online]. Available: https://doi.org/10.1007/s00464-014-3504-z

[9] T. Kim, P. M. Zimmerman, M. J. Wade, and C. A. Weiss, "The effect of delayed visual feedback on telerobotic surgery," *Surgical Endoscopy And Other Interventional Techniques*, vol. 19, no. 5, pp. 683–686, May 2005. [Online]. Available: https://doi.org/10.1007/s00464-004-8926-6

[10] H. Akasaka, K. Hakamada, H. Morohashi, T. Kanno, K. Kawashima, Y. Ebihara *et al.*, "Impact of the suboptimal communication network environment on telerobotic surgery performance and surgeon fatigue," *PLoS ONE*, vol. 17, no. 6, p. e0270039, 2022.

[11] M. Anvari *et al.*, "The impact of latency on surgical precision and task completion during robotic-assisted remote telepresence surgery," *Computer-Aided Surgery*, vol. 10, no. 2, pp. 93–99, 2005.

[12] S. Bonne *et al.*, "A digital twin framework for telesurgery in the presence of varying network quality of service," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, Mexico City, Mexico, 2022, pp. 1325–1332.

[13] G. Gonzalez, M. Balakuntala, M. Agarwal, T. Low, B. Knoth, A. W. Kirkpatrick, J. McKee, G. Hager, V. Aggarwal, Y. Xue, R. Voyles, and J. Wachs, "Asap: A semi-autonomous precise system for telesurgery during communication delays," *IEEE Transactions on Medical Robotics and Bionics*, vol. 5, no. 1, pp. 66–78, 2023.

[14] O. F. Nadjarbashi, Z. Najdovski, and S. Nahavandi, "How to predict dropped motion samples in haptic impedance devices," in *2016 International Conference on Advanced Mechatronic Systems (ICAMechS)*, 2016, pp. 459–463.

[15] H. Ishida, A. Munawar, R. H. Taylor, and P. Kazanzides, "Semi-autonomous assistance for telesurgery under communication loss," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Detroit, MI, USA, 2023, pp. 8467–8473.

[16] T. Tashiro, T. Mizoguchi, and T. Shimono, "Mutual compensation method of position and force for bilateral control systems under packet loss," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 735–740.

[17] H.-C. Hu and Y.-C. Liu, "Passivity-based control framework for task-space bilateral teleoperation with parametric uncertainty over unreliable networks," *ISA Transactions*, vol. 70, pp. 187–199, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0019057817305098

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[19] N. Madapana *et al.*, "Desk: A robotic activity dataset for dexterous surgical skills transfer to medical robots," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 6928–6934.

[20] B. Hannaford, J. Rosen, D. W. Friedman, H. King, P. Roan, L. Cheng, D. Glozman, J. Ma, S. N. Kosari, and L. White, "Raven-ii: An open platform for surgical robotics research," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 4, pp. 954–959, 2013.

[21] D. Neumuth, F. Loebe, H. Herre, and T. Neumuth, "Modeling surgical processes: A four-level translational approach," *Artificial intelligence in medicine*, vol. 51, no. 3, pp. 147–161, 2011.

[22] K. Hutchinson, Z. Li, I. Reyes, and H. Alemzadeh, "Towards surgical context inference and translation to gestures," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 6802–6809.

[23] K. Hutchinson, I. Reyes, Z. Li, and H. Alemzadeh, "Compass: a formal framework and aggregate dataset for generalized surgical procedure modeling," *International Journal of Computer Assisted Radiology and Surgery*, pp. 1–12, 2023.

[24] E. M. Ritter and D. J. Scott, "Design of a proficiency-based skills training curriculum for the fundamentals of laparoscopic surgery," *Surgical Innovation*, vol. 14, pp. 107–112, June 2007.

[25] M. Hwang *et al.*, "Applying depth-sensing to automated surgical manipulation with a da vinci robot," in *2020 International Symposium on Medical Robotics (ISMR)*, Atlanta, GA, USA, 2020, pp. 22–29.

[26] S. A. Pedram, P. Ferguson, J. Ma, E. Dutson, and J. Rosen, "Autonomous suturing via surgical robot: An algorithm for optimal selection of needle diameter, shape, and path," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, 2017, pp. 2391–2398.

[27] D. Meli, E. Tagliabue, D. Dall'Alba, and P. Fiorini, "Autonomous tissue retraction with a biomechanically informed logic based framework," in *2021 International Symposium on Medical Robotics (ISMR)*, Atlanta, GA, USA, 2021, pp. 1–7.

[28] K. Dharmarajan, W. Panitch, B. Shi, H. Huang, L. Y. Chen, T. Low, D. Fer, and K. Goldberg, "A trimodal framework for robot-assisted vascular shunt insertion when a supervising surgeon is local, remote, or unavailable," in *2023 International Symposium on Medical Robotics (ISMR)*, 2023, pp. 1–8.

[29] M. Ginesi, D. Meli, A. Roberti, N. Sansonetto, and P. Fiorini, "Autonomous task planning and situation awareness in robotic surgery," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, 2020, pp. 3144–3150.

[30] B. Kizilkaya, C. She, G. Zhao, and M. Ali Imran, "Intelligent mode-switching framework for teleoperation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 15 692–15 698.

[31] H. Saeidi, J. D. Opfermann, M. Kam, S. Raghunathan, S. Leonard, and A. Krieger, "A confidence-based shared control strategy for the smart tissue autonomous robot (star)," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. United States: IEEE, October 2018, pp. 1268–1275.

[32] E. Iovene, L. Casadio, J. Fu, F. Costa, G. Ferrigno, and E. De Momi, "Human-robot shared control for osteotomy procedure," *IEEE Transactions on Biomedical Engineering*, vol. PP, Mar. 2025, epub ahead of print. [Online]. Available: https://doi.org/10.1109/TBME.2025.3549867

[33] B. Kehoe, G. Kahn, J. Mahler, J. Kim, A. Lee, A. Lee, K. Nakagawa, S. Patil, W. D. Boyd, P. Abbeel, and K. Goldberg, "Autonomous multilateral debridement with the raven surgical robot," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1432–1439.

[34] M. Deniša, K. L. Schwaner, I. Iturrate, and T. R. Savarimuthu, "Semi-autonomous cooperative tasks in a multi-arm robotic surgical domain," in *2021 20th International Conference on Advanced Robotics (ICAR)*, 2021, pp. 134–141.

[35] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," *The International Journal of Robotics Research*, vol. 42, no. 13, pp. 1133–1184, November 2023. [Online]. Available: https://doi.org/10.1177/02783649231201196

[36] H. Zhou, Y. Jiang, S. Gao, S. Wang, P. Kazanzides, and G. S. Fischer, "Suturing tasks automation based on skills learned from demonstrations: A simulation study," in *2024 International Symposium on Medical Robotics (ISMR)*, 2024, pp. 1–8.

[37] H. H. King *et al.*, "Preliminary protocol for interoperable telesurgery," in *2009 International Conference on Advanced Robotics*, Munich, 2009.

[38] K. Weerasinghe, S. H. R. Roodabeh, K. Hutchinson, and H. Alemzadeh, "Multimodal transformers for real-time surgical activity prediction," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, Yokohama, Japan, 2024, pp. 13 323–13 330.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[40] Y. Qin, S. Feyzabadi, M. Allan, J. W. Burdick, and M. Azizian, "davincinet: Joint prediction of motion and surgical state in robot-assisted surgery," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 2921–2928.

[41] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. W. Battaglia, "Learning to simulate complex physics with graph networks," 2020. [Online]. Available: https://arxiv.org/abs/2002.09405

[42] G. Hacohen and D. Weinshall, "On the power of curriculum learning in training deep networks," 2019. [Online]. Available: https://arxiv.org/abs/1904.03626

[43] M. Ginesi, N. Sansonetto, and P. Fiorini, "Overcoming some drawbacks of dynamic movement primitives," *Robotics and Autonomous Systems*, vol. 144, p. 103844, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889021001299

[44] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Using probabilistic movement primitives in robotics," *Autonomous Robots*, vol. 42, no. 3, pp. 529–551, March 2018. [Online]. Available: https://doi.org/10.1007/s10514-017-9648-7

[45] M. Y. Seker, M. Imre, J. H. Piater, and E. Ugur, "Conditional neural movement primitives," *Robotics: Science and Systems XV*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:150384502

[46] B. Hertel and S. R. Ahmadzadeh, "Learning from successful and failed demonstrations via optimization," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 7807–7812.

[47] N. Ahmidi, L. Tao, S. Sefati, Y. Gao, C. Lea, B. B. Haro, L. Zappella, S. Khudanpur, R. Vidal, and G. D. Hager, "A dataset and benchmarks for segmentation and recognition of gestures in robotic surgery," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 9, pp. 2025–2041, Sep 2017, pMCID: PMC5559351, NIHMSID: NIHMS889842. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/28060703/

[48] A. Dutta and A. Zisserman, "The VIA annotation software for images, audio and video," in *Proceedings of the 27th ACM International Conference on Multimedia*, ser. MM '19. New York, NY, USA: ACM, 2019. [Online]. Available: https://doi.org/10.1145/3343031.3350535

[49] G. Hasslinger and O. Hohlfeld, "The gilbert-elliott model for packet loss in real time services on the internet," in *14th GI/ITG Conference - Measurement, Modelling and Evalutation of Computer and Communication Systems*, 2008, pp. 1–15.

[50] C. A. G. da Silva and C. M. Pedroso, "Packet loss characterization using cross layer information and hmm for wi-fi networks," *Sensors*, vol. 22, no. 22, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/22/8592

[51] J. J. Nielsen, I. Leyva-Mayorga, and P. Popovski, "Reliability and error burst length analysis of wireless multi-connectivity," in *2019 16th International Symposium on Wireless Communication Systems (ISWCS)*, 2019, pp. 107–111.

[52] E. Kraft, "A quaternion-based unscented kalman filter for orientation tracking," in *Sixth International Conference of Information Fusion, 2003. Proceedings of the*, vol. 1, 2003, pp. 47–54.

[53] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da vinci® surgical system," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 6434–6439.