UNIVERSITY OF SÃO PAULO
SÃO CARLOS SCHOOL OF ENGINEERING
DEPARTMENT OF MECHANICAL ENGINEERING
GRADUATE PROGRAM IN MECHANICAL ENGINEERING

Leonardo José Consoni

Adaptive Passivity Control for Haptic Transparency Enhancement of Multilateral Robotic Telerehabilitation in Virtual Reality Environments

São Carlos

2020

Leonardo José Consoni

Controle de Passividade Adaptativo para Realce de Transparência Háptica de Telerreabilitação Robótica Multilateral em Ambientes de Realidade Virtual

Tese apresentada à Escola de Engenharia de São Carlos da Universidade de São Paulo, para obtenção do título de Doutor em Ciências - Programa de Pós-Graduação em Engenharia Mecânica.

Área de concentração: Dinâmica e Mecatrônica

Orientador: Prof. Dr. Adriano Almeida Gonçalves Siqueira

VERSÃO CORRIGIDA

São Carlos

2020

Catalog card prepared by Patron Service at “Prof. Dr. Sergio Rodrigues Fontes Library” at EESC/USP

C755c

Consoni, Leonardo José
Adaptive passivity control for haptic transparency enhancement of multilateral robotic telerehabilitation in virtual reality environments. / Leonardo José Consoni; advisor Adriano Almeida Gonçalves Siqueira. -- São Carlos, 2020.

Doctoral(Thesis) – Graduate Program in Mechanical Engineering and Concentration area in Machine dynamics and systems. -- São Carlos School of Engineering, at University of São Paulo, 2020.

1. Robotics. 2. Rehabilitation. 3. Virtual reality. 4. Serious games. 5. Teleoperation. 6. Telerehabilitation. 7. Control. 8. Bilateral. 9. Passivity. 10. Transparency. 11. Internet. 12. Multiplayer. I. Título.

Prepared by Eduardo Graziosi Silva- CRB-8/8907

FOLHA DE JULGAMENTO

Candidato: Engenheiro **LEONARDO JOSÉ CONSONI.**

Título da tese: "Controle de passividade adaptativo para realce de transparência háptica de telereabilitação robótica multilateral em ambientes de realidade virtual".

Data da defesa: 14/04/2020

Comissão Julgadora	**Resultado**
Prof. Associado **Adriano Almeida Gonçalves Siqueira (Orientador)** (Escola de Engenharia de São Carlos/EESC)	APROVADO
Prof. Dr. **Teodiano Freire Bastos Filho** (Universidade Federal do Espírito Santo/UFES)	APROVADO
Prof. Dr. **Henrique Simas** (Universidade Federal de Santa Catarina/UFSC)	APROVADO
Profa. Dra. **Tatiana de Figueiredo Pereira Alves Taveira Pazelli** (Universidade Federal de São Carlos/UFSCar)	APROVADO
Prof. Dr. **Rafael Vidal Aroca** (Instituto Federal de São Paulo – IFSP)	APROVADO

Coordenador do Programa de Pós-Graduação em Engenharia Mecânica:
Prof. Associado **Carlos de Marqui Junior**

Presidente da Comissão de Pós-Graduação:
Prof. Titular **Murilo Araujo Romero**

This work is dedicated to anyone it might help someday,
either directly or indirectly, as technical or scientific contribution,
in the broader search for knowledge or solutions for a better society's quality of life.

ACKNOWLEDGEMENTS

First of all, I would like to thank my family for all the support provided during most part of my life's journey, with special attention to my parents, who constantly gave up their personal interests in order to raise my sister, my brother and me under the best possible conditions.

I would like to express my gratitude to my advisor Adriano Siqueira for his useful comments, remarks and engagement through the learning process of this doctoral thesis, as well as for placing trust in my work, pushing me to improve over previous efforts.

Besides, I wish to thank my colleagues from the Robotic Rehabilitation Laboratory, for all the technical help and collaborative work that they provided, enabling me to deliver the research results hereby presented.

I would also like to thank my friends for all the years of comradeship, accepting me and my frequent faults without lacking the honesty to point them to me, giving me the push to become a better human being.

I recognize as well the financial support provided by National Council for Scientific and Technological Development (CNPq), under grant 141058/2017-0, and Coordination for the Improvement of Higher Level Personnel (CAPES), in the form of scholarships, without which I would not be able to carry out this research.

Last but not least, I would like to thank all the people, known or anonymous, that directly or indirectly collaborated to the developments I went through over the course of my life, to whom I wish to give something in return someday.

"No man is an island,
Entire of itself,
Every man is a piece of the continent,
A part of the main.
If a clod be washed away by the sea,
Europe is the less.
As well as if a promontory were.
As well as if a manor of thy friend's
Or of thine own were:
Any man's death diminishes me,
Because I am involved in mankind,
And therefore never send to know for whom the bell tolls;
It tolls for thee."
John Donne

"Man tries to make for himself in the fashion that suits him best
a simplified and intelligible picture of the world;
he then tries to some extent to substitute this cosmos of his
for the world of experience, and thus to overcome it.
This is what the painter, the poet, the speculative philosopher,
and the natural scientist do, each in his own fashion.
Each makes this cosmos and its construction the pivot of his emotional life,
in order to find in this way the peace and security
which he cannot find in the narrow whirlpool of personal experience."
Albert Einstein

RESUMO

CONSONI, L. J. **Controle de Passividade Adaptativo para Realce de Transparência Háptica de Telerreabilitação Robótica Multilateral em Ambientes de Realidade Virtual**. 2020. 131p. Tese (Doutorado) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2020.

Nas últimas décadas, o aumento global nos casos de problemas de saúde neuromotores motivou um interesse crescente na pesquisa sobre melhoria de processos de rehabilitação utilizando robôs. Os estudos realizados abriram possibilidade para a inclusão de tecnologias auxiliares na terapia física e ocupacional, como realidade virtual e interação remota entre pacientes e terapeutas. A chamada telereabilitação, em particular, oferece o potencial de estender tratamentos, cada vez mais breves em unidades especializadas, por meio de atendimento domiciliar. A combinação da tecnologia com jogos computacionais permite que pacientes se sintam mais motivados ao treinar na companhia de amigos, familiares, ou até outros pacientes. Apesar dos benefícios verificados ou potenciais da aplicação dessas técnicas, existem desafios relacionados: instabilidades decorrentes do atraso na interação à distância, variação das características corporais de usuários e discrepância entre o nível de habilidade de participantes de uma atividade conjunta. Além disso, há poucas iniciativas no sentido de padronizar ferramentas e metodologias para sua implementação e teste, o que poderia catalisar os avanços na área. Este trabalho desenvolve uma plataforma computacional para estudo de Telereabilitação Robótica com suporte a utilização de Jogos Sérios, enquanto propõe com base na literatura disponível uma nova abordagem unificada para lidar com os principais desafios em aberto identificados. É testada a hipótese de que um mesmo conjunto de variáveis pode parametrizar a otimização de 3 subsistemas interligados: assistência automática para pacientes singulares, balanceamento de dificuldade em atividades multiusuário, e controle de estabilidade e transparência em operação remota bi ou multilateral. Ao se realizar o constante recálculdo dos parâmetros mecânicos do paciente, portanto, especula-se que o sistema pode se adaptar dinamicamente nos 3 níveis de operação, melhorando seu desempenho. Uma revisão bibliográfica prévia define os requisitos, soluções disponíveis e dificuldades ainda presentes para a implementação de tal sistema, em especial para o aspecto mais crítico da teleoperação. Testes experimentais e simulações realizados são descritos e executados, comparando abordagens já estabelecidas com as estratégias propostas. Ao fim, os resultados obtidos revelam um sucesso parcial em habilitar uma melhor interação motora entre usuários, indicando um caminho para avanço, e considerações sobre investigações necessárias e trabalhos futuros são feitas.

Palavras-chave: Robótica. Reabilitação. Telereabilitação. Realidade Virtual. Jogos Sérios. Teleoperação. Controle. Bilateral. Passividade. Transparência. *Internet*. Multijogador.

ABSTRACT

CONSONI, L. J. **Adaptive Passivity Control for Haptic Transparency Enhancement of Multilateral Robotic Telerehabilitation in Virtual Reality Environments**. 2020. 131p. Tese (Doutorado) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2020.

Over the last decades, the worldwide increase in cases of neuromotor health issues motivated a growing research interest in improving rehabilitation processes using robots. The performed studies opened the possibility to include auxiliary technologies in physical and occupational therapy, like virtual reality and remote interaction between patients and therapists. The so called telerehabilitation, in particular, offers potential extension of ever shortening periods of clinical treatment via home care. Combination of that technology with computational games allow patients to feel more motivated by exercising with friends, relatives, or even other patients. Although there are verified and potential benefits from application of those techniques, there are related challenges: instabilidades due to lag in distant interaction, variation of users' bodily characteristics and discrepancy between the skill level of participants in a joint activity. There are also still few attempts towards tooling and methodology standardization, which could lead to faster developments in the field. This work develops a computational platform for studies of Robotic Telerehabilitation with Serious Games, while proposing on the basis of the available literature a new unified approach to deal with the main open issues identified. The tested hypothesis states that the same set of variables can parameterize the optimization of 3 interconnected subsystems: automatic assistance for single patients, difficulty balancing in multi-user activities, and stability and transparency control in bi- or multilateral remote operation. Therefore, by constantly recalculating the patient's mechanical parameters, it is speculated that the system can dynamically adapt on those 3 levels of operation, improving its performance. A previous literature review narrows down the requirements, available solutions and difficulties still present for the implementation of such system, in particular for the most critical aspect of teleoperation. Performed simulations and experimental tests are described and executed, comparing more established approaches to new proposed strategies. In the end, obtained results reveal a partial success in enabling better motor interaction between users, indicating a way forward, and considerations about necessary further investigations and future work are made.

Keywords: Robotics. Rehabilitation. Virtual Reality. Serious games. Teleoperation. Telerehabilitation. Control. Bilateral. Passivity. Transparency. *Internet*. Multiplayer.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AAN	Assistance-as-needed
API	Application Programming Interface
ARM	Advanced RISC (Reduced Instruction Set Computer) Machine
BSD	Berkeley Software Distribution
CAN	Controller Area Network
CPU	Central Processing Unit
CVA	Cerebrovascular Accident
DC	Direct Current
DLL	Dynamic-link library
DoF	Degree of Freedom
sEMG	[Surface] Electromyography
EOP	Excess of Passivity
EPOS	Easy Positioning
FFB	Force Feedback
GUI	Graphical User Interface
I/O	Input and Output
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
IPC	Inter-Process Communication
ISO	International Organization for Standardization
JSON	Javascript Object Notation
LAN	Local Area Network
LQG	Linear-Quadratic-Gaussian

MIT	Massachusetts Institute of Technology
M-TDPC	Modulated Time-domain Passivity Control
NMS	Neuromusculoskeletal
OS	Operating System
PC	Personal Computer
PD	Passivity Differential
PI	Proportional-integral
PID	Proportional–integral–derivative
PM	Passivity Measurement
PREEMPT-RT	Preemptive Real-time
QoS	Quality of Service
RAM	Random Access Memory
RMS	Root Mean Square
RPC	Remote Procedure Call
RTT	Round-trip Time
SCI	Spinal Cord Injury
SEA	Series Elastic Actuator
SQL	Structured Query Language
TCP	Transmission Control Protocol
TM	Transparency Measurement
UDP	User Datagram Protocol
UML	Unified Modeling Language
USB	Universal Serial Bus
VR	Virtual Reality
XML	Extensible Markup Language
Wi-Fi	Wireless Fidelity

LIST OF SYMBOLS

t	Elapsed time (continuous)
$d\theta$	Time differential (continuous)
T	Communication delay (continuous)
Δt	Discretization period
N	Elapsed time steps (discrete)
k	k-th time step (discrete)
p	Communication delay time steps (discrete)
G	Controller transfer function/state-space matrix
Z	Plant impedance transfer function/state-space matrix
f^{int}	Human-robot interaction force/torque
K_s	Elastic actuator's spring stiffness
f_h^*	Human limb muscular active force/torque
f_h	Human limb output force/torque
M_h	Human limb intrinsic inertia
D_h	Human limb intrinsic damping
K_h	Human limb active stiffness (position tracking proportional gain)
Y_h	Human limb admittance
Z_h	Human limb impedance
x_h	Human limb position
$\dot{x}_h$	Human limb velocity
$\ddot{x}_h$	Human limb acceleration
$\hat{M}_h$	Human limb intrinsic inertia estimate
$\hat{D}_h$	Human limb intrinsic damping estimate
$\hat{K}_h$	Human limb active stiffness estimate

M_m	Motor inertia
D_m	Motor damping
R_g	Motor output gear reduction
f_r	Robot output force/torque
M_r	Robot inertia
D_r	Robot damping
Y_r	Robot admittance
x_r	Robot effector position
$\dot{x}_r$	Robot effector velocity
$\ddot{x}_r$	Robot effector acceleration
Z_r^{eq}	Robot equivalent impedance
K_r^d	Robot desired impedance
f^d	Human-robot desired interaction force/torque
f_r^d	Robot desired output force/torque
x^d	Desired output position
$\dot{x}_r^d$	Robot effector desired velocity
G^Z	Robot impedance (assistance/resistance) controller
S	Solution for matricial discrete-time Ricatti equation
G^f	Robot internal force-velocity controller
K_p	Force-velocity control proportional gain
K_i	Force-velocity control integral gain
e^x	Position error
$\hat{e}^x$	Position error estimate
e^f	Force error
J	LQG controller cost function
z	Linearized controller/filter state

u	Linearized controller/filter input
A	Linearized controller/filter state transition matrix
B	Linearized controller/filter input-state matrix
C	Linearized controller/filter observer matrix
Q	Linearized controller/filter output error weight matrix
R	Linearized controller/filter input weight matrix
ρ	Ratio between LQG cost weight matrices
w	Linearized controller/filter processing noise
v	Linearized controller/filter measurement noise
L	*Kalman* filter gain matrix
q^{out}	Generalized coordinates kinematic output
τ^{in}	Generalized input forces/torques
β	Generalized linearization coefficient
X	Generalized linearization samples matrix
x_e	Remote teleoperator-environment position
$\hat{x}_e$	Remote teleoperator-environment position estimate
$\dot{x}_e$	Remote teleoperator-environment velocity
f_e	Environment input/reaction force/torque
$\hat{f}_e$	Environment input/reaction force/torque estimate
f_e^{fb}	Environment received feedback force/torque
Y_e	Environment admittance
Z_e	Environment impedance
Y_e^h	Human-perceived environment admittance
Z_e^h	Human-perceived environment impedance
f_h^{fb}	Human received feedback force/torque
M_e	Environment inertia

D_e	Environment damping
K_e	Environment stiffness
$\hat{M}_e$	Environment inertia estimate
$\hat{D}_e$	Environment damping estimate
$\hat{K}_e$	Environment stiffness estimate
P^{in}	Input power
f^{in}	Power port input force/torque
x, x^{out}	Power port output position
$\dot{x}, \dot{x}^{out}$	Power port output velocity
E	Net/stored energy
μ, ν	Wave variables pair
μ_m	Master output wave variable
ν_m	Master input wave variable
ν_s	Slave output wave variable
μ_s	Slave input wave variable
f_m	Master side wave transformation force/torque
f_s	Slave side wave transformation force/torque
b	Wave impedance
μ^{out}	Output/sent wave variable
μ^{in}	Input/received wave variable
λ	Communication bandwidth
$\hat{\mu}^{in}$	Initial estimate of input/received wave variable
$\bar{\mu}^{in}$	Filtered input/received wave variable
γ	Wave scaling factor
ε	Wave energy excess threshold
f^{ext}	Teleoperation terminal total external force/torque

f^{fb}	Teleoperation terminal effectively applied feedback force/torque
$\hat{f}^{fb}$	Teleoperation terminal corrected remote feedback force/torque
f^{max}	Teleoperation terminal feedback force/torque absolute limit
D^c	Passivity control damping injection
f^c	Passivity control damping force/torque injection
E^{extra}	Teleoperation terminal remote input energy excess
Γ	Passivity control modulation factor
r	Remote reference state
$\hat{r}$	Remote reference state estimate
y	Delayed remote reference signal/measurement
$\hat{y}$	Remote reference signal/measurement estimate
F	Remote state prediction matrix
L^y	*Kalman* prediction gain matrix
H	Remote reference observation matrix
T_{cal}	Total calibration time period
T_{op}	Total operation time period
x^{off}	Human/player controller position offset
x^{min}	Human/player controller position lower bound
x^{max}	Human/player controller position upper bound
Δx	Human/player controller position range/amplitude
x^{player}_{game}	Game-scaled human/player controller position
x^{d}_{game}	Game-scaled desired controller position
x^{obj}_{game}	Game-scaled controlled virtual object position
$\dot{x}^{obj}_{game}$	Game-scaled controlled virtual object velocity
f^{off}	Human-robot controller interaction force/torque offset
f^{int}_{game}	Game-scaled human-robot controller interaction force/torque

f_{game}^{input}	Game-scaled human/player controller input force/torque
f_{game}^{fb}	Game-scaled remote feedback force/torque
α^x	Position scaling factor
α^f	Force scaling factor
$f^{a/r}$	Robot assistance/resistance force/torque
U^{in}	Transmitted wave signal integral
y^μ	Delayed wave signal vector
F^μ	Remote wave state prediction matrix
L^μ	*Kalman* wave state prediction gain matrix
H^μ	Wave signal observation matrix
z^e	LQG teleoperator state
L^e	LQG teleoperator's *Kalman* filter gain matrix
G^e	LQG teleoperator's controller gain matrix
y^x	Delayed remote position signal/measurement
H^x	Remote position observation matrix
y^f	Delayed remote force/torque signal/measurement
H^f	Remote force/torque observation matrix
$F^{x,f}$	Remote position and force/torque state prediction matrix
f^{norm}	Normalized absolute human-robot interaction force
$\hat{M}_{cal}^{avg}$	Average human inertia estimated during calibration period
$\hat{D}_{cal}^{avg}$	Average human damping estimated during calibration period
$\hat{K}_{cal}^{avg}$	Average human active stiffness estimated during calibration period
$\hat{M}_{op}^{avg}$	Average human inertia estimated during operation period
$\hat{D}_{op}^{avg}$	Average human damping estimated during operation period
$\hat{K}_{op}^{avg}$	Average human active stiffness estimated during operation period

CONTENTS

1 INTRODUCTION

1.1 General Context

Current worldwide population aging, combined with proliferation of other risk factors (e.g. smoking, diabetes, sedentarism and hypertension), is increasing the number of people more susceptible to or effectively affected by health issues that compromise neuromotor ability, as orthopedic injuries and neurological disorders. Their effects not only have an impact on personal *activities of daily living* (ADLs), but also result in a significant economic burden for society, due to loss of human productive capacity and additional costs related to medical treatment and rehabilitation (KREBS et al., 2008).

For instance, *cerebrovascular accident* (CVA or stroke) is already one of the three major causes of immediate or premature death in the world, killing about a third of its 15 million annual victims. It is also the leading cause of disability among adults, affecting half of the survivors. Money expenses due to CVAs were estimated in more than US$ 50 million just for the USA during 2004, and their human cost reaches about 50 million DALYs (disability-adjusted life years) globally (WORLD HEALTH ORGANIZATION, 2004).

Another serious issue, *spinal cord injury* (SCI), by its turn, have an incidence rate of 500 thousand cases each year, and in low- and middle-income countries only 15% of the impaired people have access to required assistive devices (WORLD HEALTH ORGANIZATION, 2013). Cerebral Palsy, Multiple Sclerosis and Parkinson's Disease are also among the critical age-related health problems (KREBS et al., 2008).

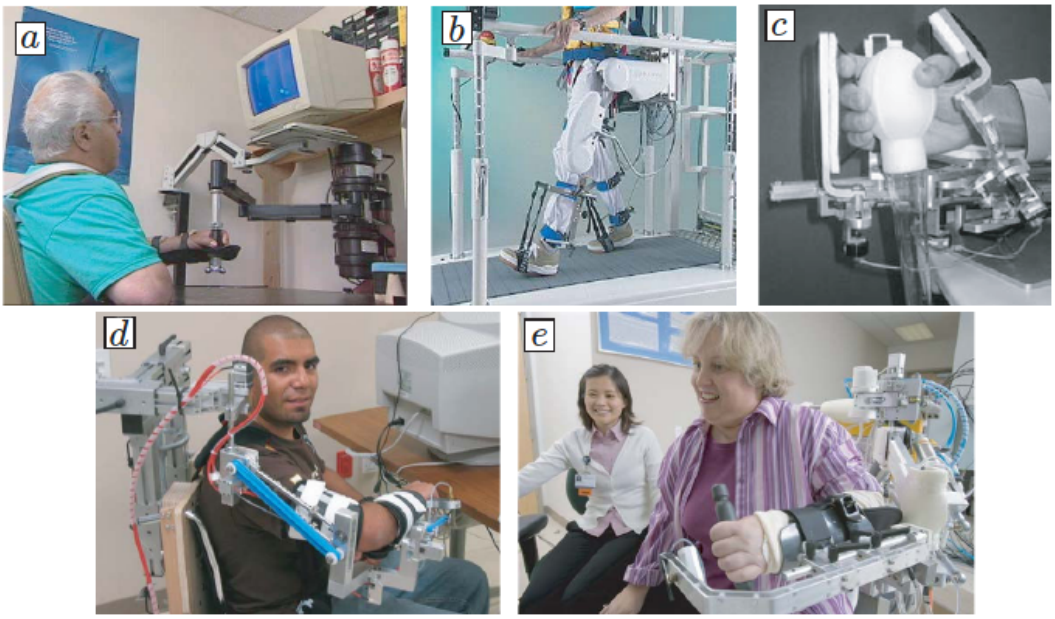
Paradoxically, as medical advances in first aid and subsequent treatments allow for an increasing survival rate and life expectancy for victims of those traumatic injuries and illnesses, that results in a raising number of survivors who suffer from motor and cognitive sequelae, whose total or partial recovery takes longer-term measures (HOGAN et al., 2006; HOGAN, 2014).

As a consequence, there is a raising demand for rehabilitation services, consuming more resources from public and private health systems, already overloaded in many countries, particularly in developing ones (WORLD HEALTH ORGANIZATION, 2004). Thus, there is a pressing need for innovative procedures that could replace or supplement conventional physical and occupational therapy, capable of recovering patients in a quicker and less costly way, mitigating the economic and social impact of the aforementioned diseases (CARIGNAN; KREBS, 2006; KREBS et al., 2008; LI et al., 2015).

1.2 Robotic Rehabilitation

In light of those facts and recent technological advancements, research centers have shown an increasing interest in rehabilitation robotics over the last decades (KREBS et al., 2008; MARCHAL-CRESPO; REINKENSMEYER, 2009; CAO et al., 2014). Formerly used only as an assistive technology to compensate for physical limitations, custom robots – as controllable end-effectors, orthoses or exoskeletons, as exemplified in Fig. 1 – started being applied, with relative success, to replace therapist's manual support and guidance of patient's limbs during prescribed exercises, working as an interface between those two (CAO et al., 2014; HOGAN, 2014; DONATI et al., 2016).

Figure 1 – Examples of robotic therapy devices using different types of assistance-based control algorithms: (a) MIT-MANUS, (b) Lokomat, (c) HWARD, (d) T-WREX, and (e) Pneu-WREX.



Source: Modified from Marchal-Crespo and Reinkensmeyer (2009)

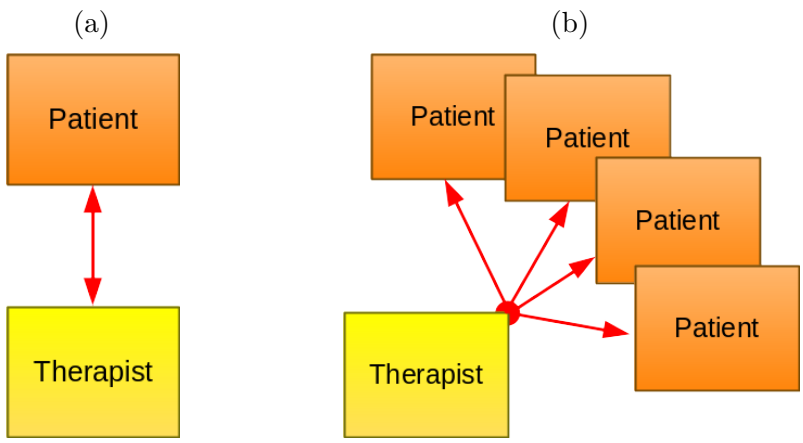
The direct benefits provided by this intermediary include requiring less professionals to conduct a treatment session and achieving more movement repetitions in the same time, as robots do not get tired. Additionally, patient's motion performance – effort, regularity, amplitude – evolution can be assessed more objectively by registering robot's sensors measurements, and assistance or resistance forces can be applied by its actuators, as needed, more precisely (KREBS et al., 2008; HOGAN, 2014; SHAHBAZI et al., 2016). Apart from the mechanical preadjustment possible for passive devices, those actively controlled can even have their software parameters customized in advance for each user's anthropometric and biological characteristics (JUTINICO et al., 2017; PÉREZ-IBARRA et al., 2019; PEÑA et al., 2019).

1.3 Telerehabilitation

Robot-aided neurorehabilitation alone, however, does not present a solution to the entire issue. The immediate advantages of higher treatment intensity and reduced clinician workload are limited by cost-containment (initially worsened by acquisition of robotic equipment), low availability of adequate medical facilities and patient's mobility difficulties due to severe impairment, which tend to be compensated by ever shortening hospitalization times (CARIGNAN; KREBS, 2006; GORSIC; NOVAK, 2016), compromising recovery.

But having a computerized robotic system incorporated in the process enables additional approaches. With a reliable direct link or *Internet* connection, two or more rehabilitation participants could remotely interact, visually and mechanically, even when separated by long distances. That feature could be used by a therapist to provide home-based medical care (KARIME et al., 2015; LI et al., 2015; MINGE et al., 2017; AL-MAHMOOD; AGYEMAN, 2018) to a patient (Fig. 2a) or supervise multiple individuals taking part in group activities (Fig. 2b). This usage of telecommunication technologies for medical purposes is named telemedicine, which in this case is more specifically defined as telerehabilitation (CARIGNAN; KREBS, 2006; LI et al., 2015).

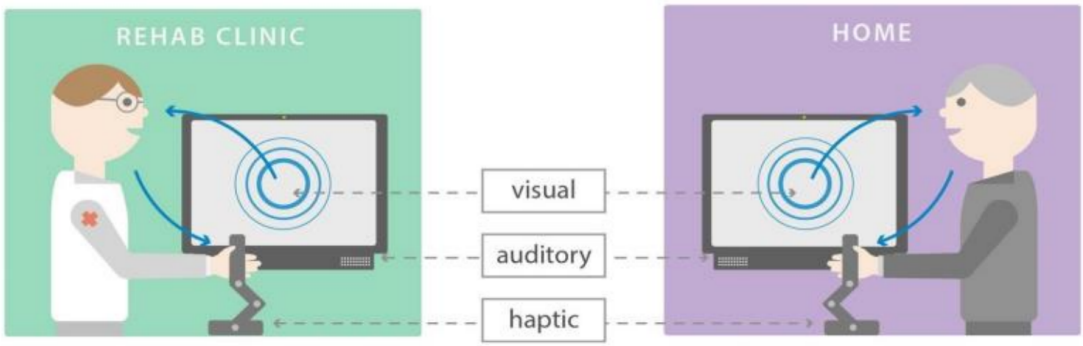
Figure 2 – Representation of potential uses for telerehabilitation technology: (a) remotely assisted home care and (b) therapist-coordinated group therapy.



Source: The author, adapted from Carignan and Krebs (2006)

Performing therapy over distance presents its own set of challenges, though, to be overcome in order to provide a more cost-effective solution to physicians and impaired people in constraining conditions. Even if some sort of interaction could be offered through audio and video transmission, a complete representation of a human counterpart also requires contact (haptic) force reflection, as represented by Fig. 3. In this context that motor interaction is called bilateral control (ANDERSON; SPONG, 1989; CARIGNAN; KREBS, 2006) – or multilateral control, when force sense is shared among more than two operators over the network (KAWASE; MIYOSHI; TERASHIMA, 2016).

Figure 3 – Representation of audiovisual and haptic (bilateral) interaction between robotic telerehabilitation participants.



Source: Modified from Minge et al. (2017)

Like any kind of remote communication, teleoperation is bound by its own nature to time delays and finite bandwidth, which may disrupt transparency (faithful impedance reproduction) and lead to instability inside feedback loops (NIEMEYER; SLOTINE, 2004; HOKAYEM; SPONG, 2006; RODRÍGUEZ-SEDA, 2015). Besides, diagnostics could not rely on close examination, but would depend mostly on recorded data and periodical surveys (BAUR; SCHÄTTIN, et al., 2018).

1.4 Serious Games for Rehabilitation

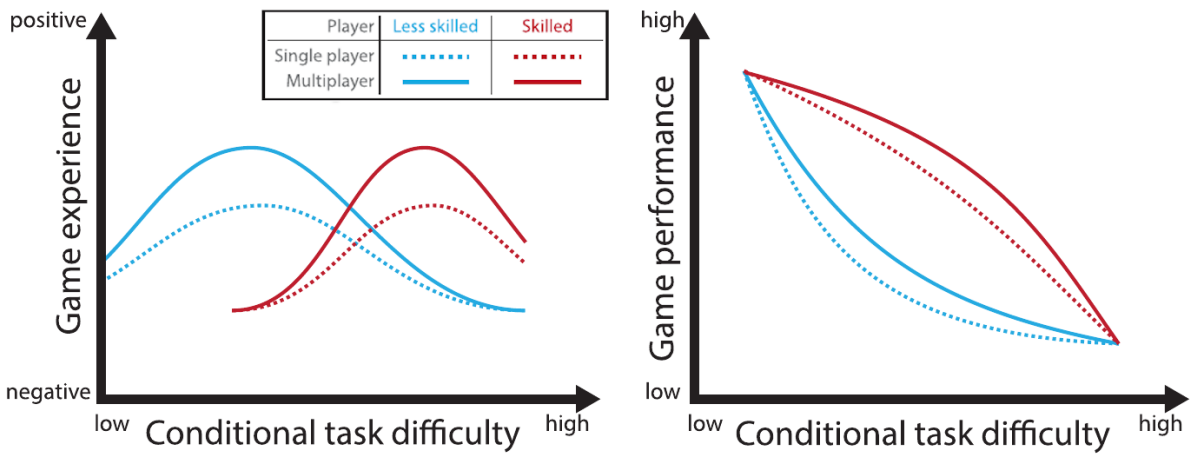
Another obstacle for neuromotor rehabilitation that should be addressed is the motivation issue: the prescribed exercises are typically based on lengthy and monotonous repetition, which may quickly become tedious, discouraging or even being uncomfortable to impaired patients, whose commitment plays a key role in proper recovery (HOGAN et al., 2006; HOGAN, 2014). Gamification can help as a way to rate and reward treated subjects for their achievements, thus keeping interest and engagement throughout the treatment (GONÇALVES et al., 2014; BAUR; SCHÄTTIN, et al., 2018). Those games intended for goals beyond player's enjoyment, like functional training, are classified as *serious games* (ANDRADE et al., 2013).

When computers with graphical capabilities are available, *virtual reality* (VR) serves as a tool for serious games by presenting motor tasks and performance assessment in an intuitive audiovisual format, instigating and reacting to user's actions (ANDRADE et al., 2013; GONÇALVES et al., 2014; PRAHM et al., 2017). Also, their algorithms can integrate automatic difficulty adaptation to increase training intensity as the player improves (BAUR; WOLF, et al., 2017; DARZI; GORŠIČ; NOVAK, 2017). The use of immersive VR devices even has potential to mitigate the sensation of pain and fatigue (MATSANGIDOU et al., 2019) or reduce cognitive load (WENK et al., 2019).

On telerehabilitation systems, video-games would work to enhance remote human

interaction through agonistic or antagonistic interpersonal gameplay, which has a known potential to be more motivating than single-player ones (ANDRADE et al., 2013; NOVAK et al., 2014; GORSIC; NOVAK, 2016; BAUR; SCHÄTTIN, et al., 2018). Fig. 4 shows how multiplayer game modes (solid lines), when compared to single-player ones (dotted lines), influence both game experience (left) and performance (right) for less skilled (light blue) and more skilled (dark red) players. The benefit of multiplayer is present for all skill levels and at all conditional task difficulties.

Figure 4 – Comparison of participant experience and performance in single-player and multiplayer game modes.



Source: Modified from Baur, Schättin, et al. (2018)

In this case, challenge adaptation shall work not only for patient evolution, but also for load balancing between participants with distinct skill or impairment levels, to keep them stimulated (NOVAK et al., 2014; DARZI; GORŠIČ; NOVAK, 2017; BAUR; WOLF, et al., 2017).

1.5 Motivation

Despite the several multiplayer difficulty matching algorithms proposed in literature (BAUR; SCHÄTTIN, et al., 2018) for adjusting virtual elements or player conditions, based on e.g. game score (NOVAK et al., 2014; BAUR; WOLF, et al., 2017) or human bio-signals (DARZI; GORŠIČ; NOVAK, 2017), none of those investigated seem to tackle the bilateral (or multilateral, by extension) control aspect of telerehabilitation.

The ultimate goal in seeking patient participation is to promote relearning of the affected limb movements (HOGAN, 2014), and the feedback haptic forces would not influence as intended if they cannot be reflected with high fidelity (WILDENBEEST; ABBINK; SCHORSCH, 2013), as when communication is under the effect of delays or data losses. That may have not been a serious issue to most previous studies due to the fact that they were performed in local networks (NOVAK et al., 2014; BAUR; WOLF, et al.,

2017; DARZI; GORŠIČ; NOVAK, 2017; GORŠIČ; DARZI; NOVAK, 2017) or simulated environments (KAWASE; MIYOSHI; TERASHIMA, 2016).

Besides, estimations of patient effort – used for both assistance and game challenge adaptation – which are dependent on additional equipment (DARZI; GORŠIČ; NOVAK, 2017; PEÑA et al., 2019) are less suitable for low-cost seeking applications of remote/home care (GORSIC; NOVAK, 2016; DARZI; GORŠIČ; NOVAK, 2017). It is desirable that this type of information could be obtained from very portable devices or directly from control data.

On the other hand, the robotic teleoperation algorithms generally lack that online update feature of gameplay balancing, with its controller model parameters being set beforehand, manually (MUNIR; BOOK, 2002) or via some kind of offline identification process (MIYOSHI; TERASIMA; BUSS, 2006; ATASHZAR; SHAHBAZI, et al., 2017). Even when some adjustment to physical changes in local and remote environment happens, it consists of discrete switching between a limited amount of operation modes (RODRÍGUEZ-SEDA, 2015; JUTINICO et al., 2017).

When dealing with the more realistic condition of time-varying delay and dynamics during multilateral robotic therapy, unifying assistance-level (local control and user effort assessment), teleoperation-level (bilateral control modeling and reflection stabilization) and game-level (performance measurement and challenge balancing) adaptation might combine their advantages and deliver a simpler solution for assistance, stable transparency and social interaction to promote motor learning.

1.6 Objectives

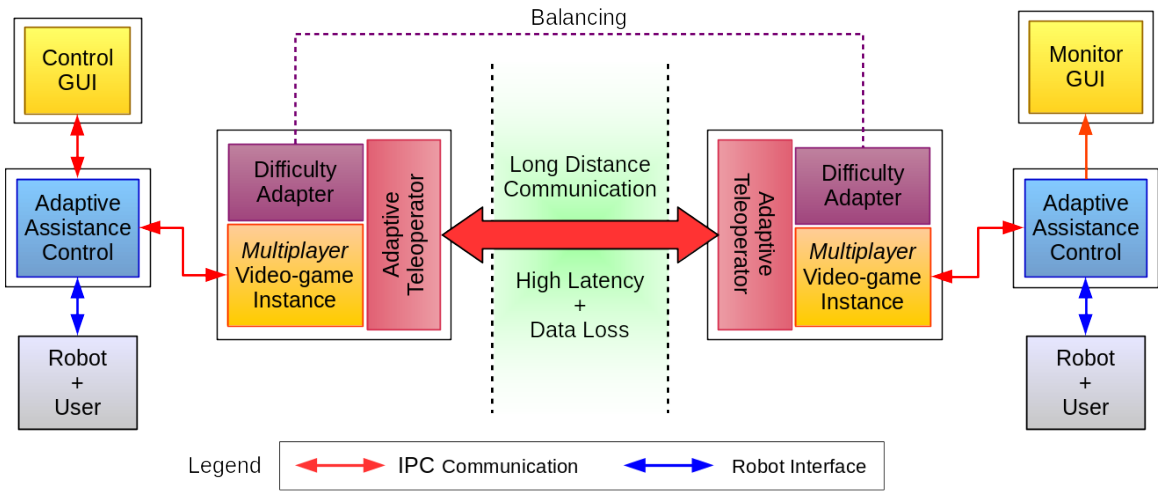
1.6.1 General Goal

With that integration hypothesis in mind, this work proposes the development of a custom software platform for application of robot-aided therapy with multiple available hardware, virtual reality games and telerehabilitation features, as illustrated by Fig. 5 (black borders delimit different hardware pieces, colored borders delimit software components), differentiating itself from known solutions by providing defined interfaces for streamlined modification or replacement of logical components.

1.6.2 Work Scope and Specific Goals

Although the entire system presents 3 levels of adaptation, the main focus here is on the teleoperation (bi or multilateral control) problem, but with special attention to how its solutions relate to the other subcomponents. As presented in Fig. 6, by interpreting them as functions of the same set of parameters, optimizing one block would provide meaningful data for adapting the entire system, which allows for more straightforward

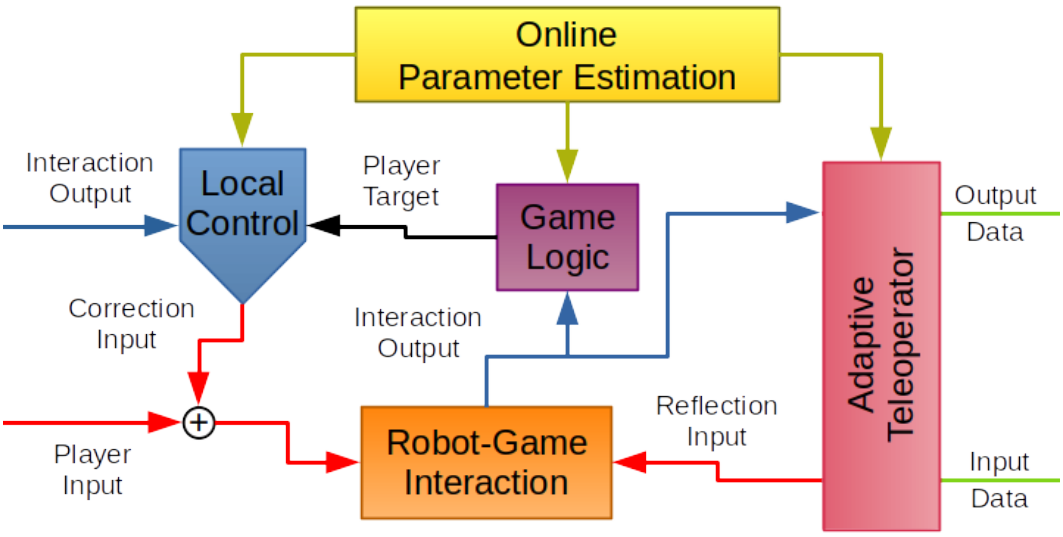
Figure 5 – Top-level representation of proposed platform and interchangeable subsystems.



Source: The author, modified from Consoni (2017)

integration.

Figure 6 – Diagram of unified parameter estimation logic for the 3 adaptation subsystems.



Source: The author

As proof of concept, a fully functional implementation shall be able to:

- Offer enough flexibility to operate with an arbitrary number of users, different robotic devices, physical therapy modalities and control algorithms, without core modifications;
- Handle latency effects during multiplayer telerehabilitation game matches, ensuring stable force exchange even under heavy communication lag constraints, keeping it mostly playable throughout the entire session;

- Dynamically identify operators' mechanical behaviour in a way that works for online tuning of robot assistance, bilateral/multilateral control parameters and per-player game difficulty;
- Collect meaningful data about user performance, that could aid a therapist's evaluation in a real patient treatment case;
- Enable effective telerehabilitation, displaying, according to some objective criteria, that conditions for proper remote motor teaching/learning – namely stable transparency – are ensured;
- Additionally, provide adequate documentation of platform's internals and available interfaces, so that other researchers have enough information to further develop it.

1.7 Justification and Intended Purpose

The proposed platform is designed that way in order to provide a unified test and operation basis for robotic rehabilitation technologies. This is not limited to the research hereby presented, and ideally subsequent projects will be able to utilize it to develop hardware or software improvements and customizations as new modules.

1.8 Intended Contribution

In addition to the developed computational tools, the main contributions for the field of knowledge in question are demonstrating that:

- The 3 aforementioned subsystems could be made dependent on the same set of monitored variables, so that adjustment based on them helps encourage human participation, optimize the teleoperation stability-transparency trade-off and balance competitive, collaborative and cooperative activities;
- With online adaptation of teleoperation algorithm's parameters, one could achieve better performance than what is currently verified for solutions in literature.

1.9 Text Structure

The following text is summarized and organized as follows: Chapter 2 presents a literature review of the main challenges for each subarea of game-based robotic telerehabilitation and the record of most relevant solutions found up until now; Chapter 3 describes the solution proposed in this work and the tooling implemented for deploying it; Chapter 4 shows the testing procedures and the obtained results, discussing them in light of the theory and previous expectations; Finally, Chapter 5 concludes the work, evaluating its achievements and pointing potential new developments.

2 LITERATURE REVIEW

For later reference purposes, a review of its theoretical foundations and previous implementations is necessary. This chapter attempts to gather the most relevant sources on current challenges and available solutions for each addressed issue.

It should be noted that it is outside the scope of this project to implement all the possible solutions for the proposed platform, but that knowledge is required in order to substantiate design decisions for the intended application and future developments.

2.1 Rehabilitation Robotics

Industrial robots can be designed under the assumption that they will always move inside a restricted and unobstructed workspace. Thus, motion accuracy tends to be their major performance requirement, what is generally achieved by building these kinds of equipment with heavy and rigid mechanical parts, to minimize vibration, and applying simple PID (*proportional–integral–derivative*) controllers for desired motion tracking (HOGAN, 2014).

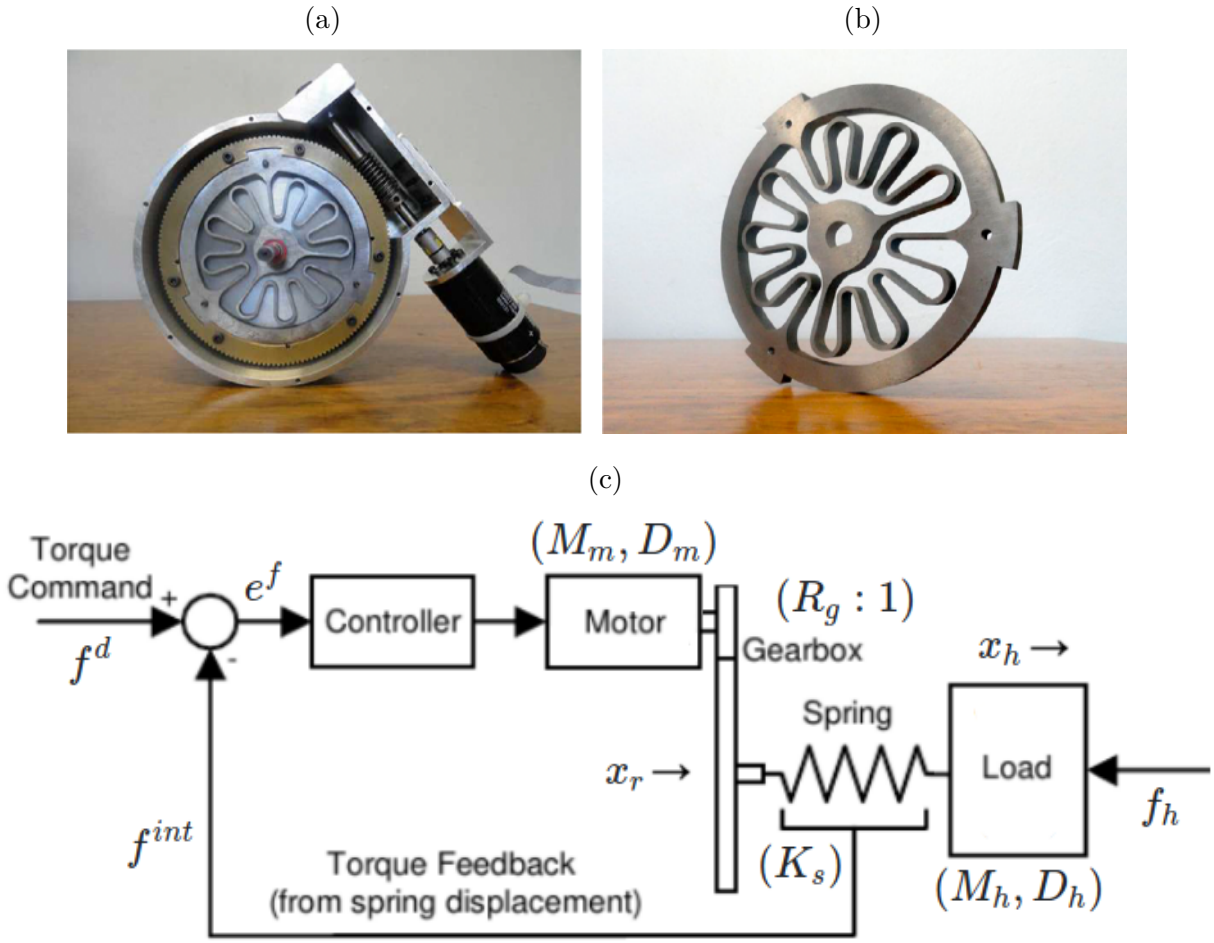
In contrast, devices designed for interaction with human subjects (for augmentation, assistance, evaluation, or rehabilitation) should not be built using the same structural or control principles of an industrial machine. For safety and usability reasons, such robots shall be lightweight and backdrivable, and are not intended to enforce any trajectory on a user's opposing movements, as the potentially harmful – especially for a patient – interaction forces need to be taken into account (ROY et al., 2009; HOGAN, 2014).

2.1.1 *Series Elastic Actuators*

A widespread architecture for human-machine haptic interfaces is the *series elastic actuator* (SEA) (WYETH, 2006), characterized by a spring-like component placed between motor drive and load carrier (Fig. 7). Fig. 7c shows how, instead of rigid coupling, human effector movement x_h deviates from robot position output x_r, causing deformation of the elastic element that links them. The additional piece serves mainly two purposes: giving the actuator an intrinsic lower rigidity, for stable contact behaviour, and being part of an interaction force f^{int} sensor, since its known average stiffness K_s and measurable deformation would be related by Hooke's Law (DOS SANTOS; CAURIN; SIQUEIRA, 2017; JUTINICO et al., 2017):

$$f^{int} = K_s(x_r - x_h) = -f_h + M_h\ddot{x}_h + D_h\dot{x}_h. \tag{2.1}$$

Figure 7 – (a,b) Rotary SEA full assembly and elastic element (c) SEA topology diagram.



Source: (a,b) Dos Santos, Caurin, and Siqueira (2017) (c) modified from Wyeth (2006)

For operation, that extra complexity – from looser coupling of human (M_h, D_h) and robot (M_r, D_r) dynamics – requires that SEA controllers resort to force-velocity internal loops in order to avoid effects of the mechanism's intrinsic friction and achieve better performance (WYETH, 2006; DOS SANTOS; CAURIN; SIQUEIRA, 2017). Jutinico et al. (2017) handle that by modeling the motor and gearbox ($R_g : 1$ reduction ratio) set as a decoupled pure velocity $\dot{x}_r$ source, with roughly proportional force f^{int} output:

$$f^{int} \approx f_r = M_r \ddot{x}_r + D_r \dot{x}_r, \quad \text{where} \quad M_r = M_m R_g^2, \; D_r = D_m R_g^2. \tag{2.2}$$

2.1.2 *Adaptive Assistance*

The usual goal of either robotic or traditional neuromotor therapy is not simply giving up position control in favor of the individual's actions, but finding a safe compromise between those two, to simultaneously demand strength and coordinate movements from the affected limb. The matching between intentional muscle activation and actually performed motion stimulates reestablishment of nerve synapses (neuroplasticity), and the lack of

this combination prevents rehabilitation and leads to a slacking condition (HOGAN et al., 2006; HOGAN, 2014; NORMAN; LOBO-PRAT; REINKENSMEYER, 2017).

In order to achieve that, control strategies for therapeutic robots are mostly based on the *assistance-as-needed* (AAN) principle: it should help as little as possible to roughly guide reference tracking, while inciting the patient to complete that by itself (KREBS et al., 2008; MARCHAL-CRESPO; REINKENSMEYER, 2009; CAO et al., 2014). But the amount of rehabilitation effort patients are able to make may vary greatly, as over the course of a single therapy session their motion performance could decrease significantly due to the accentuated effects of fatigue on impaired individuals (KREBS et al., 2008; CAO et al., 2014), so the assistance level should be adaptable to that evolution.

2.1.3 *Impedance Control*

In more general terms, AAN involves dynamically changing the relationship between trajectory x^d following error and desired assistance force f^d, much like a variable impedance (HOGAN, 2014). For a robot to perform that kind of task when dealing with the unknown properties of an environment such as a human operator, Hogan (1985) proposed what is called impedance control. This technique allows even structurally rigid actuators to have a desired soft behaviour (MARCHAL-CRESPO; REINKENSMEYER, 2009), by e.g. modulating their virtual (apparent) stiffness K_v and damping B_v:

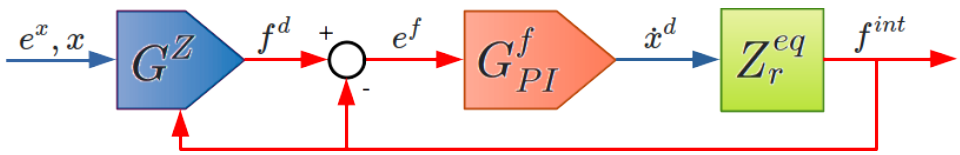
$$f^d = K_v e^x - B_v \dot{x}, \quad \text{where} \quad e^x = x^d - x, \quad x = x_h. \tag{2.3}$$

In a SEA, f^d works as setpoint for f^{int}, regulated e.g. via internal *proportional-integral* (PI) controller G^f loop (DOS SANTOS; CAURIN; SIQUEIRA, 2017; JUTINICO et al., 2017), with output velocity reference $\dot{x}^d$ for the equivalent robot impedance Z_r^{eq}:

$$\dot{x}^d = \dot{x}_r^d = K_p(f^d - f^{int}) + K_i \int_0^t (f^d - f^{int}) d\theta, \quad \text{where} \quad f^{int} \approx Z_r^{eq} \dot{x}_r. \tag{2.4}$$

Fig. 8 representation uses red for effort (e.g. force, torque) signals and blue for flow (e.g. velocity, position) ones, pattern that is maintained throughout this document.

Figure 8 – Diagram of cascaded position/impedance and force control for SEAs.



Source: The author, adapted from Dos Santos, Caurin, and Siqueira (2017)

Appropriate values for impedance controller G^Z have to be defined according to additional criteria, like position tracking performance, positive work (time integral of

output power), game task score (GONÇALVES et al., 2014; PEREZ-IBARRA; SIQUEIRA; KREBS, 2015; PÉREZ-IBARRA et al., 2019), respiration rate or muscular activation through *surface electromyography* (sEMG) (MARCHAL-CRESPO; REINKENSMEYER, 2009; CAO et al., 2014; DARZI; GORŠIČ; NOVAK, 2017).

2.1.3.1 Optimal and robust control

Instead of explicitly setting K_v and B_v or arbitrary update heuristics, the modulation of robot assistance forces could be optimized by *linear–quadratic–gaussian* (LQG) regulators (DOS SANTOS; SIQUEIRA, 2016; CONSONI; PEÑA; SIQUEIRA, 2018).

With it, the simultaneous and opposing goals of reducing output state z, relative to output error e, and control input u are balanced through minimization of a quadratic cost J function (over N operation time steps) (SKOGESTAD; POSTLETHWAITE, 2005). In its "cheap" configuration, the cost weights Q and R are symmetrical matrices whose relative ratio may be manually adjusted by a scalar factor ρ:

$$J = \sum_{k=0}^{N} \left(z_k^T Q z_k + u_k^T R u_k \right), \quad \text{where} \quad Q = C'C, \quad R = \rho I \ (\rho > 0). \tag{2.5}$$

LQG requires a linear model for controlled plant observation, which is the equivalent of a *Kalman* filter (FARAGHER, 2012). State-space matrices A, B and C determine the relationships between internal state z, input u and output error estimate $\hat{e}$ variables. The mathematical representation could be simplified, as the algorithm accounts for unstructured (stochastic) uncertainties w (processing noise) and v (measurement noise):

$$z_{k+1} = A z_k + B u_k + w_k \ , \quad -\hat{e}_k^x = C z_k + v_k. \tag{2.6}$$

In the case of observing output from a mechanical admittance, inertia M, damping D and stiffness K compose the system matrices that update position error $\hat{e}^x$ from previous state and resulting force input $\sum f$, with discretization interval Δt:

$$u = \sum f, \quad e^x = x^d - x \ , \quad A = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ \frac{-K}{M} & \frac{-D}{M} & 1 \end{bmatrix}, \ B = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{M} \end{bmatrix}, \ C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}. \tag{2.7}$$

Even if patient's passive dynamics are considered, its generated forces f_h are usually left as unmodeled disturbance (DOS SANTOS; CAURIN; SIQUEIRA, 2017; JUTINICO et al., 2017), as they are not directly measurable and may vary greatly, including random unintentional reactions caused by neurological disorders (spasticity) (HOGAN, 2014; ATASHZAR; SAXENA, et al., 2014).

The full discrete optimal controller equations (for any sampling period Δt) are a combination of state-space plant model, internal state z observer (*Kalman* filter) and compensator (feedback gain) G^Z, as in:

$$z_{k+1} = Az_k + Bu_k + L_{k+1}(x_k - x_k^d - C(Az_k + Bu_k)), \quad \text{where} \quad u_k = -G^Z z_k. \tag{2.8}$$

The *Kalman* correction gain L is recalculated for every step as described in Faragher (2012), while G^Z comes from matricial solution to discrete-time algebraic Ricatti equation:

$$\begin{gathered} G^Z = (B^T SB + R)^{-1} B^T SA, \quad \text{where} \\ A^T SA - S - (A^T SB)(R + B^T SB)^{-1}(B^T SA) + Q = 0. \end{gathered} \tag{2.9}$$

If the actual plant presents significant nonlinearities (e.g. variations of M or D) inside its working range, an iterative variant of the LQG controller (TODOROV; LI, 2005) may be applied. In that case, the feedback gains shall also be recalculated for every new set of state-space matrices linearized on a different point.

Although not intrinsically robust (bounded transfer function gain), the LQG method leaves room for stability increase via the LTR (*Loop Transfer Recovery*) process, that for "cheap" control consists of lowering ρ to near 0 (SKOGESTAD; POSTLETHWAITE, 2005). Alternatively, a more robust solution like H_∞ synthesis (DOS SANTOS; CAURIN; SIQUEIRA, 2017) or Markovian chains (JUTINICO et al., 2017) could be applied.

2.1.3.2 User effort estimation

One more advanced way to handle man-machine mechanical interactions is modeling the entire system as a single plant to be controlled (HOGAN, 2014), with online estimation of person's active impedance and input force, in order to have direct knowledge of user's movement intention (TAGLIAPIETRA et al., 2015) and operate within narrower stability margins for better performance.

Processing sEMG measurements with *neuromusculoskeletal* (NMS) dynamic models of the human body might offer a reliable approximation of joint torques (SETH et al., 2011). However, most equations present a computational cost too high for real-time processing (DURANDAU; FARINA; SARTORI, 2018). Even if it could be mitigated by neural network approximation (PEÑA et al., 2019), proper sEMG acquisition requires equipment and technical expertise not always available in domestic environments (TAGLIAPIETRA et al., 2015). Also, adjusting parameters for each patient's particular anatomical properties would be a complex and error prone procedure, making it not much scalable for clinical usage.

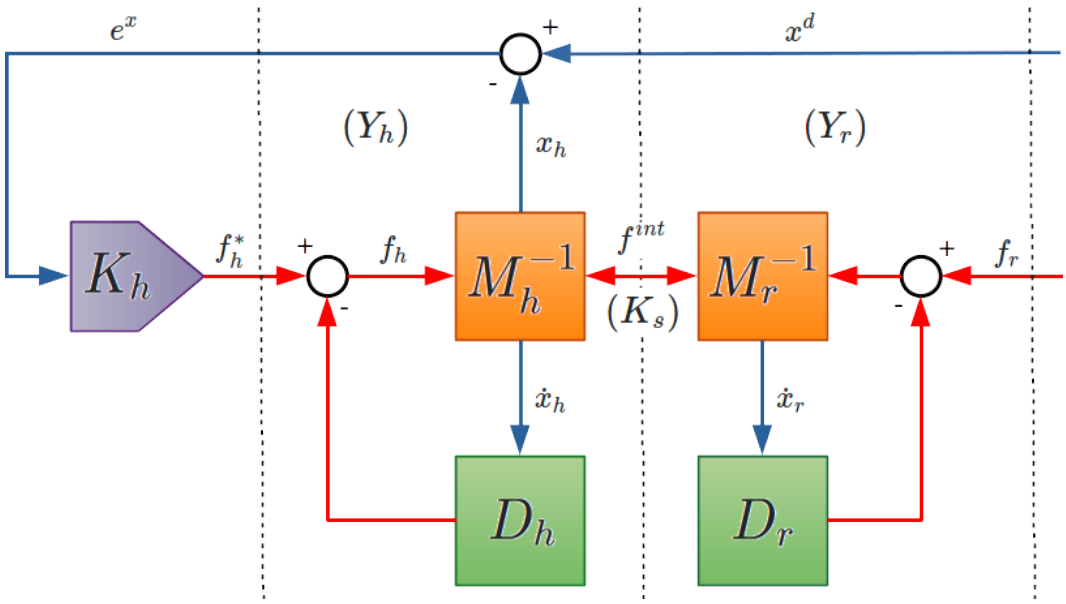
Simpler approaches may be used if further assumptions are made about user's behaviour: since the motor task will generally have a known reference x^d, it is common

to consider output human force f_h as the result of a muscular impedance controller (MIYOSHI; TERASIMA; BUSS, 2006; PÉREZ-IBARRA et al., 2019; DOS SANTOS; SIQUEIRA, 2016) as (Eq. 2.3), with stiffness K_h and damping D_h:

$$f^{int} = f_h - M_h\ddot{x}, \quad \text{where} \quad f_h = f_h^* - D_h\dot{x}_h, \quad f_h^* = K_h e^x. \tag{2.10}$$

That way, patient's contribution can be indirectly obtained – as a factor of the human model's active input f_h^* – from already available kinematic data. Fig. 9 shows how human force could be assumed proportional to reference x^d tracking error, while limb admittance (Y_h) has inertia-damping behaviour like the robotic actuator (Y_r). As an advantage, the same values of K_h or f_h^* serve as effort quantification, enough to differentiate among several physical conditions, for possible game challenge adaptation (PÉREZ-IBARRA et al., 2019; DOS SANTOS; SIQUEIRA, 2016).

Figure 9 – User's effort and impedance estimation model.



Source: The author

Mefoued, Mohammed, and Amirat (2013), Dos Santos and Siqueira (2016), Jutinico et al. (2017) and Consoni, Peña, and Siqueira (2018) use the least squares method to some extent for fitting m plant parameters β to a set of n input τ^{in} (e.g. force, torque) and output q^{out} (e.g. position, velocity, acceleration) samples, recorded on an offline identification phase. In matrix form, that is a computationally efficient approach to linearization that allow even for real-time processing during operation itself, if required:

$$\tau^{in} \approx \sum_{m}^{i} \beta_i q_i^{out} \implies \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_m \end{bmatrix} = (X^T X)^{-1} X^T \begin{bmatrix} \tau_1^{in} \\ \vdots \\ \tau_n^{in} \end{bmatrix}, \quad X = \begin{bmatrix} q_{1,1}^{out} & \dots & q_{1,m}^{out} \\ \vdots & \dots & \vdots \\ q_{n,1}^{out} & \dots & q_{n,m}^{out} \end{bmatrix}. \tag{2.11}$$

A generic implementation of an impedance control algorithm – considering end-effector reference trajectory tracking – is presented in Code 2.1. Internal subroutines PROCESS_MODEL, MOD_IMPEDANCE, and FORCE_CONTROL_STEP may have multiple implementations, like any of the solutions presented throughout this section.

```
function RUN_CONTROL_STEP( position, reference,
                           velocity, acceleration,
                           interaction_force ):

 position_error = reference - position

 user_effort = PROCESS_MODEL( position_error, velocity,
                              acceleration, interaction_force )

 desired_force = MOD_IMPEDANCE( position_error, velocity,
                                acceleration, user_effort )

 return FORCE_CONTROL_STEP( interaction_force, desired_force )
endfunction
```

Code 2.1 – Pseudocode describing a simplified impedance modulation algorithm.

2.2 Robotic Telerehabilitation

On first examination, it might seems possible that automatic AAN would suffice to take care of a patient training outside of a clinical facility, like in home rehabilitation, as the robot plays the role of a virtual therapist. However, it is virtually impossible to design an adaptation law that replicates the expertise of a skilled professional (ATASHZAR; POLUSHIN; PATEL, 2013). In practice, robotic training still requires external human corrective intervention and monitoring in what may be called a supervised operation mode (SHAHBAZI et al., 2016; MINGE et al., 2017).

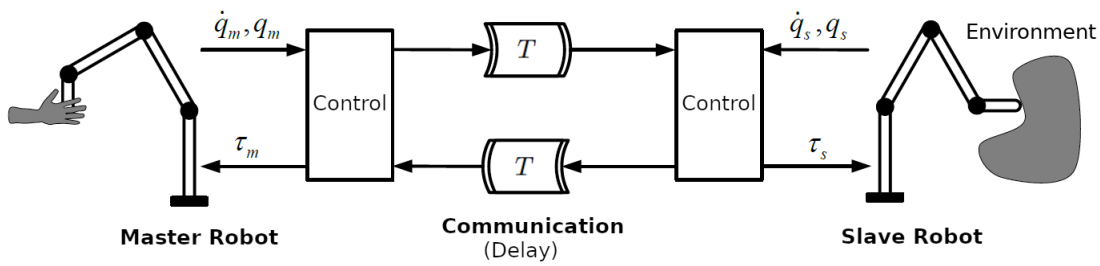
For telerehabilitation, when the remote therapist is more limited in the ways he or she can interact, the active device should not simply execute supervisor's commands, but offer some local semi-autonomous control, ensuring that the distant partner participation input is properly reproduced on both directions, keeping the benefits of interpersonal interaction for learning, while maintaining the stability and safety of the movements carried out (WILDENBEEST; ABBINK; SCHORSCH, 2013; SHAHBAZI et al., 2016; ATASHZAR; SHAHBAZI, et al., 2017).

2.3 Bilateral Teleoperation

Telerehabilitation is a particular use case for teleoperation, so that the same definitions, theorems and control algorithms apply for both. According to Anderson and Spong (1989) and Carignan and Krebs (2006) teleoperator classification, working configurations could be either unilateral or bilateral, depending on whether or not feedback forces are remotely transmitted (reflected).

As in Niemeyer and Slotine (2004), bilateral control terminals are named "master" and "slave" when only the first has a proactive role on manipulation. Fig. 10 describes forward and backward data channels, slave force feedback to master trajectory input, and the doubled delay T effect.

Figure 10 – Representation of master-slave bilateral robotic teleoperation.



Source: Modified from Niemeyer and Slotine (2004)

Kawase, Miyoshi, and Terashima (2016) introduce the concept of multilateral telecontrol, but as a combination of interconnected bilateral channels, so the theory essentially does not change. Those force reflection architectures are the focus of this work, as they enable haptic interaction and motor assistance between patients and therapists.

2.3.1 *Strategies for Bilateral Control*

Impedance modulation can ensure stability for digital control when time discretization steps are small enough to approximate the continuous case. When processors cannot guarantee a minimum update frequency or measurements are considerably delayed, instabilities appear. The second issue becomes unavoidable when data transmissions over long distances are part of the processing loop, due to intrinsic network latency.

The available literature contains various algorithms intended for delay compensation on remote robot operation (HOKAYEM; SPONG, 2006), but many present some key disadvantage, like: need for full system modeling; reliance on accurate delay estimation, which is difficult to achieve; or stability loss under variable lag (jitter), characteristic of ubiquitous *Internet* communications (RODRÍGUEZ-SEDA; LEE; SPONG, 2009). Thus, methods that provide better overall performance, robustness and easier implementation are prioritized in this review.

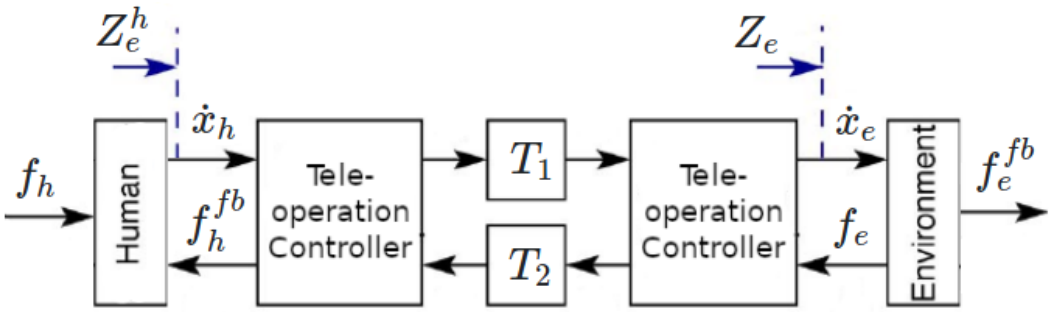
Also, being able to cope with delays well over 200 milliseconds is not a hard requirement for selected solutions, as transmissions of that quality are already considered inadequate for real-time applications (SUZNJEVIC; SALDANA, 2015). Though, that does not mean a huge distance restriction, as with an optical fiber link *round-trip time* (RTT) – total latency for sending a message and receiving its reply – could remain below 100 ms even between places about 10000 km apart (CONSONI; SIQUEIRA; KREBS, 2017).

2.3.1.1 Transparency

Apart from effector position control, in human-operated remote interaction, getting to "feel" the environment accurately is also an important requirement, as it helps the user with decision-making and generalized learning (WILDENBEEST; ABBINK; SCHORSCH, 2013). Thus, sacrificing optimal tracking performance to enable more faithful dynamics reproduction on the opposing side of communication is a justifiable design decision.

In more technical terms, a teleoperation system shall be transparent by displaying to manipulators a mechanical impedance Z_e^h or admittance Y_e^h equivalent to the environmental ones (Z_e or Y_e, respectively). Following Fig. 11, ideal transparency would mean that the system between Z_e^h and Z_e is virtually inexistant (such that $f_e^{fb} = f_h$ and $f_h^{fb} = f_e$), considering human perception tolerances (HIRCHE; BUSS, 2012). An admittance representation is achieved by symply inverting the variables flow direction.

Figure 11 – Representation of impedance perception from local user's perspective.



Source: Modified from Hirche and Buss (2012)

2.3.1.2 Passivity

Most teleoperation applications involve interaction with passive environments, i.e. ones that do not react adding energy to the whole system, but only conserving or dissipating it. So, in order to reflect that behaviour, common bilateral control techniques (NIEMEYER; SLOTINE, 2004; HOKAYEM; SPONG, 2006) operate under the assumption that energy returned to any terminal will never be greater than an $E(0)$ previously stored in them plus what is inserted through the m input ports. In other words, the net work

performed by input forces on master side would be always positive:

$$E(t) \;=\; \int_0^t \sum_m^i P_i^{in} d\theta \;=\; \int_0^t \sum_m^i f_i^{in} \dot{x}_i^{out} d\theta + E(0) \;\geq\; 0. \tag{2.12}$$

That "weak" formulation of passivity theorem states that the coupling of two subsystems remains stable while it is ensured that each separate part is energetically passive (ATASHZAR; SHAHBAZI, et al., 2017), which is true for human limb dynamics but may be invalidated by delayed contact response.

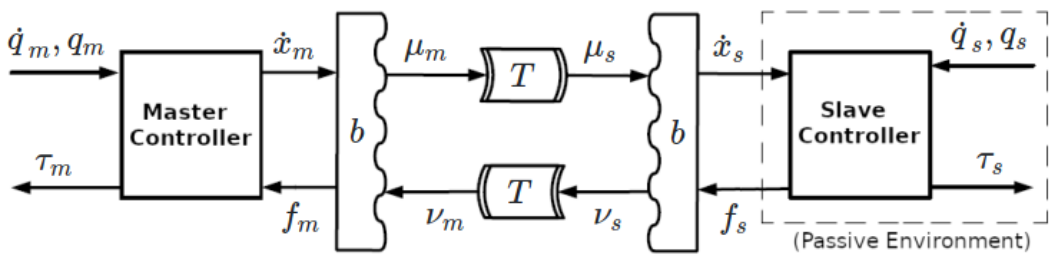
2.3.1.3 Wave variables

Based on scattering transformations, wave variables are an encoding and decoding scheme for velocity $(\dot{x}_m, \dot{x}_s)$ and force (f_m, f_s) values transmitted over network in bilateral teleoperation, modulated by a free impedance factor b (HOKAYEM; SPONG, 2006):

$$\mu_m = \frac{b\dot{x}_m + f_m}{\sqrt{2b}}, \quad \nu_s = \frac{b\dot{x}_s - f_s}{\sqrt{2b}}, \quad \mu_s = \frac{b\dot{x}_s + f_s}{\sqrt{2b}}, \quad \nu_m = \frac{b\dot{x}_m - f_m}{\sqrt{2b}} \tag{2.13}$$

Theoretically, they assure "weak" (local) passivity (Eq. 2.12) for any given target system, as in Fig. 12, under any finite constant transmission latency T, with no need for plant modeling. Assuming some additional constraints, manipulation stability could be also guaranteed for discrete updates (NIEMEYER; SLOTINE, 2004; RODRÍGUEZ-SEDA; LEE; SPONG, 2009), which is the case with this work.

Figure 12 – Wave variables transformations and transmission in bilateral teleoperation.



Source: Modified from Niemeyer and Slotine (2004)

Those variables μ and ν form a complementary pair of forward (output) and backward (input) signals – respectively from a master point of view, and the opposite for its slave – from which the interaction forces in Fig. 11 can be inferred:

$$\begin{aligned} \mu_s(t) &= \mu_m(t-T), \\ \nu_m(t) &= \nu_s(t-T) \end{aligned} \quad \Longrightarrow \quad \begin{aligned} f_e(t) &= f_s(t) = \sqrt{2b}\mu_s(t-T) - b\dot{x}_e(t), \\ f_h^{fb}(t) &= f_m(t) = b\dot{x}_h(t) - \sqrt{2b}\nu_m(t-T). \end{aligned} \tag{2.14}$$

In practice, by combining effort and flow values, wave variables work as virtual energy exchanges between corresponding actuators, which naturally limit motion: as communication delays get larger, the master becomes stiffer to user input, in order to keep it stable; when latency falls, controller compliance raises, turning synchronization more transparent. That trade-off sensitivity is regulated via the wave impedance b.

In wave coordinates, the passivity condition for the communication channel is represented by an inequality between terminal input μ^{in} and output μ^{out} integrals

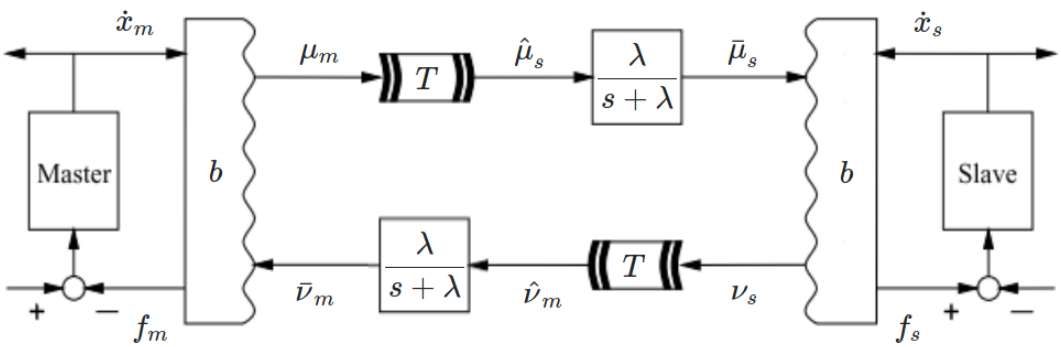
$$\sum_{i=0}^{2}\int_0^t \frac{1}{2}(\mu_i^{in})^2 d\theta \leq \sum_{i=0}^{2}\int_0^t \frac{1}{2}(\mu_i^{out})^2 d\theta + E(0), \quad \text{where} \quad \begin{array}{l} \mu_1^{in} = \mu_s,\ \mu_2^{in} = \nu_m, \\ \mu_1^{out} = \mu_m,\ \mu_2^{out} = \nu_s. \end{array} \tag{2.15}$$

One unintended effect of those transformations are signal oscillations called wave reflections, which intensify with increasing disparities between master and slave mechanical dynamics (e.g. when a joystick is used to remotely control a big robot). In order to minimize their effects without more complex system modeling and impedance matching, Niemeyer and Slotine (2004) propose integration of a low-pass incoming wave filter for initial estimate $\hat{\mu}^{in}$, whose output $\bar{\mu}^{in}$ depends on communication bandwidth λ:

$$\frac{\bar{\mu}^{in}(s)}{\hat{\mu}^{in}(s)} = \frac{\lambda}{s+\lambda} \implies \text{discrete}: \bar{\mu}_{k+1}^{in} = \frac{(2-\lambda)\bar{\mu}_k^{in} + \lambda(\hat{\mu}_{k+1}^{in} + \hat{\mu}_k^{in})}{2+\lambda}. \tag{2.16}$$

Its usage dismisses the need for extra position controllers in the slave, allowing for a symmetrical setup with direct force output as feedback ($f_m = f_h^{fb}$,$f_s = f_e^{fb}$) on both sides, as displayed in Fig. 13.

Figure 13 – Wave variables reflection low-pass filtering.



Source: Modified from Niemeyer and Slotine (2004)

Besides, matching of velocities does not prevent actuator positions x to drift apart from each other over time, due to packet losses or integration inaccuracies. To fix that, subsequent versions of the originally proposed algorithm started to add position feedback

x^d for calculation of a correction component $\Delta\mu$ that ensures trajectory tracking without violating the passivity requirements for the resulting wave μ^{in} (NIEMEYER; SLOTINE, 2004; RODRÍGUEZ-SEDA; LEE; SPONG, 2009):

$$\mu^{in} = \bar{\mu}^{in} + \Delta\mu, \qquad \Delta\mu = \begin{cases} 0 & \text{if } d\bar{\mu}^{in} < 0, \text{ with } d = x - x^d \\ -\sqrt{2b}\lambda d & \text{if } d\bar{\mu}^{in} > 0 \text{ and } \sqrt{2b}\lambda|d| < |\bar{\mu}^{in}| \\ -\bar{\mu}^{in} & \text{if } d\bar{\mu}^{in} > 0 \text{ and } \sqrt{2b}\lambda|d| > |\bar{\mu}^{in}| \end{cases}. \tag{2.17}$$

However, an unaddressed source of instability appears when network latency oscillations are at play, as they deform the wave signal and hinder the reconstruction of flow-effort variables. Munir and Book (2002) deals with it using *Kalman* filters while Rodríguez-Seda (2015) relies on the extra transmission of quadratic wave integrals signals in order to check the teleoperator energy level and scale $\hat{\mu}^{in}$ down as needed:

$$\hat{\mu}_i^{in}(t) = \gamma\mu_i^{out}(t-T), \;\; \gamma = \begin{cases} 1 & \text{if } E_i \geq \varepsilon_i \\ \frac{2\varepsilon_i^2 E_i^2}{E_i^4+\varepsilon_i^4} & \text{otherwise} \end{cases}, \;\; E_i = \int_0^{t-T} (\mu_i^{out})^2 d\theta - \int_0^t (\hat{\mu}_i^{in})^2 d\theta \tag{2.18}$$

It can be seen that b is a free parameter for wave transformations, used to optimize the stability-transparency compromise to external conditions: small values work better for free motion, while with big ones hard contact is more properly handled (NIEMEYER; SLOTINE, 2004). Although touch sensors could trigger discrete switching and synchronization of wave impedance values (RODRÍGUEZ-SEDA, 2015), online adaptation to remote user behaviour would probably require some continuous identification process.

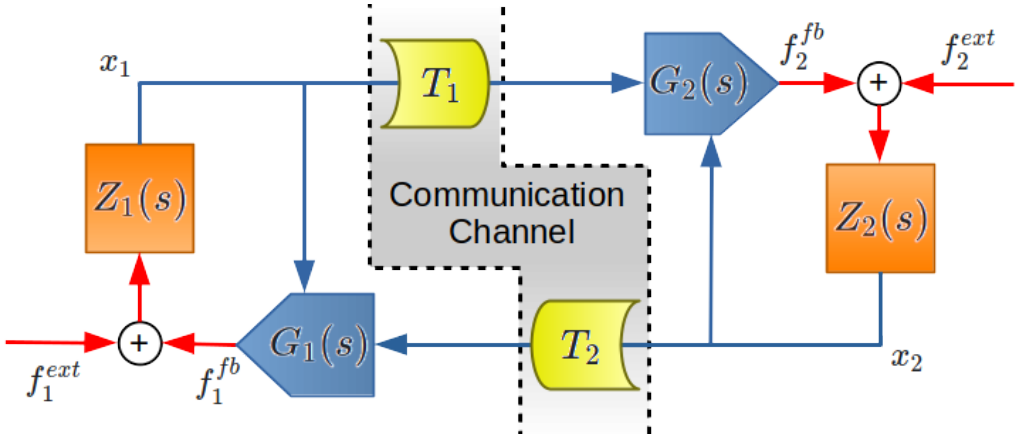
2.3.1.4 Small-gain criterion

Despite some works (CARIGNAN; KREBS, 2006; KAWAI et al., 2017) proposing and testing wave scattering usage in telerehabilitation, Atashzar, Polushin, and Patel (2012) question that approach, as activities with external coordination or augmentation assistance represent interaction with non-passive environments, thus contradicting the original assumption. Enforcing per-subsystem local passivity condition (Eq. 2.12) may cancel out any remote assistance input, leading to poor transparency and defeating the purpose of a telesupervisor (ATASHZAR; SHAHBAZI, et al., 2017).

According to Miyoshi, Terasima, and Buss (2006), that is an unnecessary and even unattainable requirement, from a control perspective. Alternatively, it would be sufficient for teleoperation stability, under any constant delay value, to ensure that the whole plant Z and controller G closed loop, shown in Fig. 14, respects a small-gain criterion:

$$\|Z_1(s)G_1(s)\|_\infty \|Z_2(s)G_2(s)\|_\infty \leq 1. \tag{2.19}$$

Figure 14 – Generalized model for bilateral teleoperation feedback loop.



Source: The Author, adapted from Miyoshi, Terasima, and Buss (2006)

In order to fulfill that, the authors add to the system wave filters that modify its transfer function's H_∞ norm. Based on the same principle, Atashzar, Polushin, and Patel (2012) propose a received force feedback f^{fb} saturation scheme that also deals with time-varying delays, whose effectiveness is sustained through simulations and mathematically demonstrated by Lanini et al. (2015):

$$f^{fb} = \begin{cases} \hat{f}^{fb}, & \text{if } |\hat{f}^{fb}| \leq f^{max} \\ \frac{\hat{f}^{fb}}{|\hat{f}^{fb}|} \cdot f^{max}, & \text{otherwise} \end{cases} \quad , \text{ where } \quad f^{max}(t) \leq \|Z_h(s)\|_\infty \cdot \sup_{\theta \in [t-T,t]} |\dot{x}_e(\theta)|. \tag{2.20}$$

Saturated result f^{fb} is the effectively applied interaction response, actually delivered to the local operator, which only differs from the originally reflected/calculated $\hat{f}^{fb}$ when the latter goes beyond a dynamic safety threshold f^{max}, maximizing reflection transparency. The force limit depends on the lower bound of user impedance Z_h and varies with the greatest received value of remote velocity $\dot{x}_e$ inside a time window of duration T. Unlike wave variables, stabilization of nonpassivity sources still occurs when $T \to 0$.

One notable disadvantage of this method in relation to wave variables resides in its requirement for a model of human limb, even if time-invariant, without which the feedback limitation could not be calculated properly. Yet, that would have no additional cost if making use of the results from effort and dynamics identification for local control (as later described in Section 3.3).

2.3.1.5 Energy filters

Although force saturation enables stable communication under variable latency and without the conservative approach of passivating subsystems individually, the teleoperator may be subject to undesired incoherences of feedback direction, a phenomenon known as

task inversion, when phase shift is high enough, effectively inhibiting the remote therapy (ATASHZAR; POLUSHIN; PATEL, 2013).

An alternative solution less impacted by activation thresholds is *time-domain passivity control* (TDPC) (ATASHZAR; SHAHBAZI, et al., 2017). Its theoretical foundation is the "strong" form of passivity theorem, stating that when there is a nonpassive remote terminal in a haptics-enabled system, the closed-loop can still remain stable if the energy dissipation capacity D_h of local dynamics can compensate for the additional positive work of f^{fb}. That way, partner reflected assistance forces only have to be alleviated by artificial damping force f^c injection when the extra net energy E^{extra} is above 0:

$$f^{fb} = \hat{f}^{fb} - f^c, \quad f^c = \begin{cases} D^c\dot{x} & \text{if } |D^c\dot{x}| \leq |\hat{f}^{fb}| \\ \hat{f}^{fb} & \text{if } |D^c\dot{x}| > |\hat{f}^{fb}| \end{cases}, \quad D^c = \begin{cases} 0 & \text{if } E^{extra} \leq 0 \\ \frac{E^{extra}}{\dot{x}^2\Delta t} & \text{if } E^{extra} > 0 \end{cases}. \tag{2.21}$$

Atashzar, Shahbazi, et al. (2017) apply a modulation factor Γ in order to give a different weight to most recent *passivity differential* (PD) sample, mixing properties from time-domain (more transparent, delayed response and overshoots) and power-domain (smoother response, less transparent) E^{extra} observers for better overall performance:

$$\begin{aligned} E^{extra}_{k+1} &= \Gamma(E^{extra}_k - f^c_k\dot{x}_k\Delta t) + PD_{k+1}, \\ \text{where} \quad PD &= (f^{fb}\dot{x} - D_h\dot{x}^2)\Delta t, \quad 0 \leq \Gamma \leq 1. \end{aligned} \tag{2.22}$$

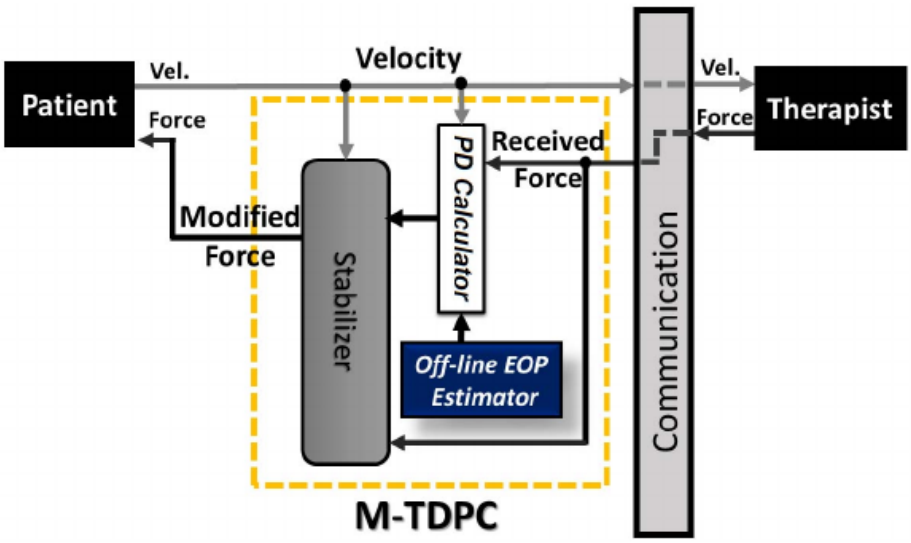
It is important to note that Miyoshi, Terasima, and Buss (2006), Atashzar, Polushin, and Patel (2012) and Atashzar, Shahbazi, et al. (2017) develop their solutions for a master-slave setup, with unilateral stabilization (Fig. 15), in which the therapist is taken only as a provider of assistance/resistance forces, directly reflected to patient's side. Also, patient operator's *excess of passivity* (EOP) – used for H_∞ norm calculation or stabilization damping injection – is estimated during an offline calibration phase with a relaxed user (taking lower bound damping properties for safety reasons).

Even so, the same principles are expected to hold for a symmetrical control (force feedback and filtering on both terminals) setup with online EOP identification.

2.3.1.6 Signal prediction

Another way to understand transparency is as simultaneous matching of force and positional information on both teleoperation actuators (WILDENBEEST; ABBINK; SCHORSCH, 2013), which is a more straightforward performance target than keeping impedance/admittance consistency. With that in mind, being able to predict ahead-of-time remote signals is useful to compensate the natural destabilization and distortion effects of telecommunication lag on haptic perception.

Figure 15 – Diagram of unilateral stabilization setup, using modulated TDPC (M-TDPC).



Source: Atashzar, Shahbazi, et al. (2017)

However, simple extrapolation is prone to sharp changes which might harm user experience. A usual solution is the application of Smith predictors (MUNIR; BOOK, 2002), but that would require knowledge of the opposing terminal's transfer function. For a potentially optimal and safer estimation $\hat{y}$, Larsen et al. (2002) describe incorporation of delayed readings in discrete *Kalman* filters via future state r projection (average latency T may be calculated as half RTT), followed by retroactive correction gain L^y when the backward in time corresponding measurement y is received:

$$\begin{aligned} \hat{r}_{k+p} &= F_k^{k+p}\hat{r}_k + L_{k+1}^y(y_k - H\hat{r}_k) \\ \hat{y}_{k+p} &= H\hat{r}_{k+p} \end{aligned} \quad , \quad \text{where} \quad p = \left\lfloor \frac{T}{\Delta t} \right\rfloor . \tag{2.23}$$

2.4 Multiplayer Serious Games Development

Special purpose games whose design requirements are beyond entertainment, as when applied in physical therapy, are called *serious games*. Prior studies show evidence that the integration of virtual games in robotic rehabilitation is able to increase user motivation and engagement in exercise (ANDRADE et al., 2013), a decisive factor in motor recovery (NOVAK et al., 2014). More immersive forms of *virtual reality* (VR) could even help reduce cognitive load in spacial tasks (WENK et al., 2019) or pain and fatigue sensation during exercise (MATSANGIDOU et al., 2019).

2.4.1 *Development Tools*

From Novak et al. (2014), one can infer that features that are appealing in conventional video-games could not be essential or even desirable for serious games. Large

amounts of graphical and sound effects that enhance commercial titles, for instance, could distract or disturb the player during task execution (BAUR; SCHÄTTIN, et al., 2018). This issue is aggravated in the case of neurological rehabilitation like post stroke treatment, where the patient's impairment makes any excess of provided information lead to confusion and consequent inability to fulfill the virtual goals.

Therefore, it is not necessary that employed tools offer the most advanced game rendering technologies, and factors like ease of use and availability of learning material could be prioritized when choosing one. Even so, Cowan and Kapralos (2014) verify that well-know modern game engines like *Unity3D*[1] and *Unreal*[2] end up being the most picked by serious games developers, as they present an integrated environment for coding, visual interface design, physics processing, etc.

2.4.2 *Networking*

When considering a multi-interface and multi-device computer system, apart from a physical data transmission link, the software itself shall make use of some remote *inter-process communication* (IPC) method, capable of establishing not only loopback (internal) connections but between different machines. BSD sockets end up among the most widespread IPC solutions for programming (HALL, 2011), operating over *Ethernet* or *Wi-Fi* networks using the *Internet protocol* (IP).

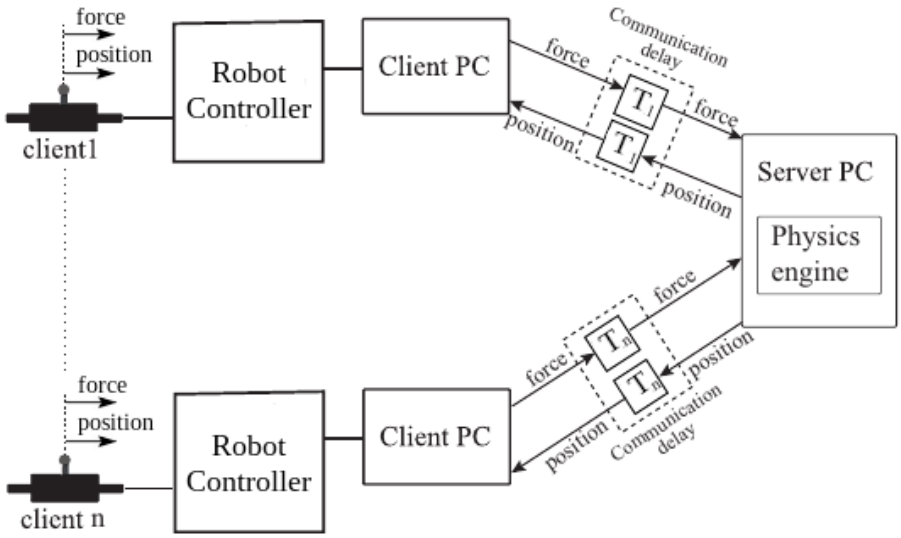
It is important to note that many articles in this field emphasize usage of UDP (*User Datagram Protocol*) over TCP (*Transmission Control Protocol*) for real-time online applications (MUNIR; BOOK, 2002; HOKAYEM; SPONG, 2006; KAWASE; MIYOSHI; TERASHIMA, 2016). Despite lacking message integrity guarantees from the latter, smaller overhead and no packet batching of UDP (HALL, 2011) are desirable, since most correction algorithms suffer greater performance degradation due to latency and signal deformation than eventual network data loss (RODRÍGUEZ-SEDA; LEE; SPONG, 2009).

Regarding the interconnection layout, Kawase, Miyoshi, and Terashima (2016) adopt a client-server model for multilateral teleoperation games. Fig. 16 shows how a central server PC node simulates the virtual environment where a number n of player clients interact and are sinchronized remotely.

Centralizing all physics calculations in an authoritative server helps simplifying the synchronization code at the cost of extra inter-client communication delay and consequent input lag. The technique of client-side prediction (FIEDLER, 2014; CONSONI; SIQUEIRA; KREBS, 2017) may be used to improve immediate response to player actions through local auxiliary simulations interpolating received server states.

[1] https://unity3d.com
[2] https://www.unrealengine.com/what-is-unreal-engine-4

Figure 16 – Client-server model for multilateral teleoperation games



Source: Modified from Kawase, Miyoshi, and Terashima (2016)

2.4.3 *Gameplay Modes*

Besides the underlying infrastructure, it is necessary to comprehend the effects of different types of gameplay, in order to prescribe specific motor tasks and design the game around them. The task characteristics and the players behaviour define the behavioral characteristics of the multiplayer mode.

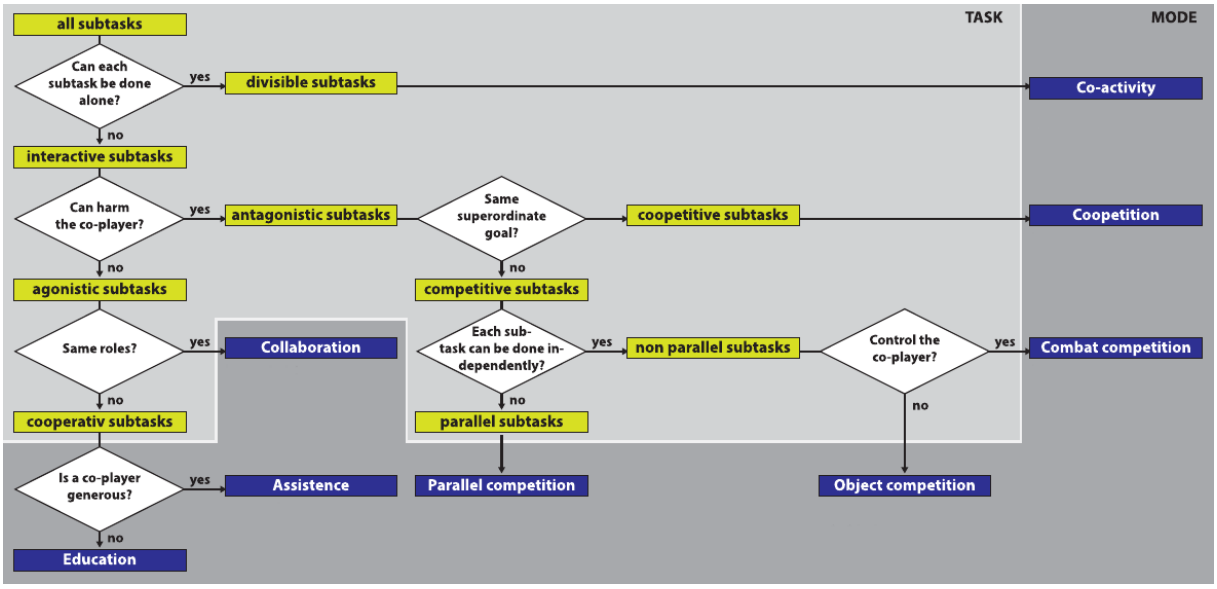
Following the terminology adopted in (JARRASSÉ; CHARALAMBOUS; BURDET, 2012), corresponding to Fig. 17, multiplayer games involving interaction between participants can be either co-active, competitive, cooperative or collaborative, which approximate to free, resistive, passive and assistive multilateral operation, respectively.

As shown in Novak et al. (2014), a proper matchup between patients' personality and gameplay mode has the potential to increase enjoyment and participation in two-player games over *local area network* (LAN), while Baur, Schättin, et al. (2018) state that purely remote interaction has no impact in motivation in comparison with cases when players are in the same room.

2.4.4 *Difficulty Adaptation and Balancing*

Providing a rehabilitation patient with a challenge well suited for his/her impairment level is more effective in increasing willingness to deal with harder tasks – a key factor in neuromotor recovery – than a multiplayer setting alone (BAUR; WOLF, et al., 2017). Not only the game goal itself shall be achievable, but flow models also require that players display similar skill levels in order to prevent boredom, frustration or stress, particularly for competitive activities (BAUR; SCHÄTTIN, et al., 2018).

Figure 17 – Determination of multiplayer modes, based on the task characteristics.



Source: Modified from Baur, Schättin, et al. (2018)

Adaptation to co-participants performance may be conducted in basically two ways: modification of game-level mechanics and visual hints or controller-level assistance/resistance. The first one causes global changes that are apparent to everyone involved, which can be embarrassing for worse performing players, while the second – possible via robotic interfaces – allow for more individual, subtle and fine-grained adjustments, more useful for balancing (BAUR; SCHÄTTIN, et al., 2018).

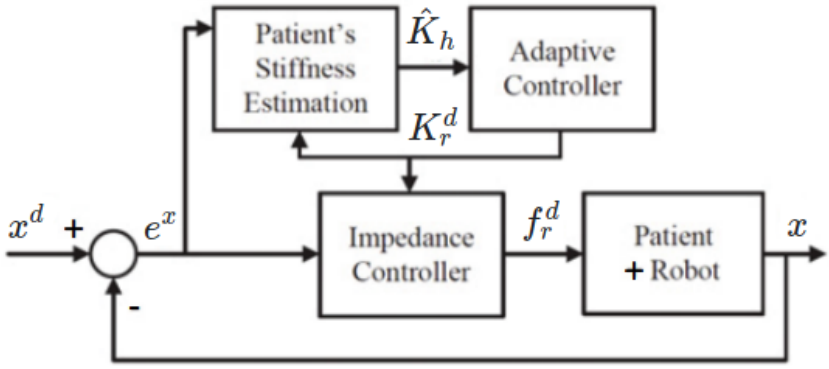
For instance, Perez-Ibarra, Siqueira, and Krebs (2015) and Pérez-Ibarra et al. (2019) apply modulation of orthosis impedance K_r as AAN difficulty optimization during a therapeutic video-game. Following Fig. 18, desired position x^d tracking error e^x is used to estimate human active stiffness (proportional gain) $\hat{K}_h$ and combined with adaptable target robot stiffness K_r^d to calculate the control action f_r^d. Although being a single-player activity, the idea of unified mechanical interaction models for robot and gameplay control could be naturally extended to teleoperation transparent stabilization, making use of the aforementioned techniques.

2.5 Closing Remarks

The literature review presented here was intended not only to find state-of-the-art solutions for the 3 adaptation levels (Fig. 5) of robotic telerehabilitation gaming, but also to expose areas of intersection among them, where the proposed integrated optimization process (Fig. 6) could be implemented.

For all subsystems, some model of mechanical impedance/admittance (Eq. 2.7) is considered for at least one approach, whose actual parameter values (e.g. K, D, M)

Figure 18 – Adaptive impedance control based on patient stiffness estimation and position reference tracking error.



Source: Modified from Perez-Ibarra, Siqueira, and Krebs (2015)

estimation is most commonly an offline linearization process. For tunning those parameters to time-varying operation conditions, identification may be performed online with well-conditioned initial values and prevention of numerical anomalies.

In order to validate that, we concentrate on the more complex teleoperator problem – involving the main performance bottleneck of communication delays. The tested solutions are developed with user effort and gameplay balancing models in mind, for later extension. A state-space and adaptive variant of some studied bilateral control algorithms, whose features are summarized in Tab. 1, shall be implemented and tested for initial feasibility evaluation.

Table 1 – Teleoperation algorithms feature comparison.

Algorithm	**Modeling Required**	**Main Advantage**	**Main Disadvantage**
Wave Variables	Wave impedance (b) tunning	Simple implementation	Enforces local passivity
H_∞ **Norm Control**	Whole loop plant (Z_1, Z_2)	Full loop robustness	Complex modeling
Force Saturation	Local impedance (Z_h)	Favours transparency	Eventual task inversion
Passivity Control	Dissipation capacity (D_h)	Manages global passivity	Accommodation window
Signal Prediction	State Prediction (F)	Seeks ideal transparency	Unbounded output

Source: The author

3 MATERIALS AND METHODS

In this chapter, description of the most important groundwork for the proposed project is presented. Developed software applications, hardware tools and strategies applied to address the previously stated problem are detailed.

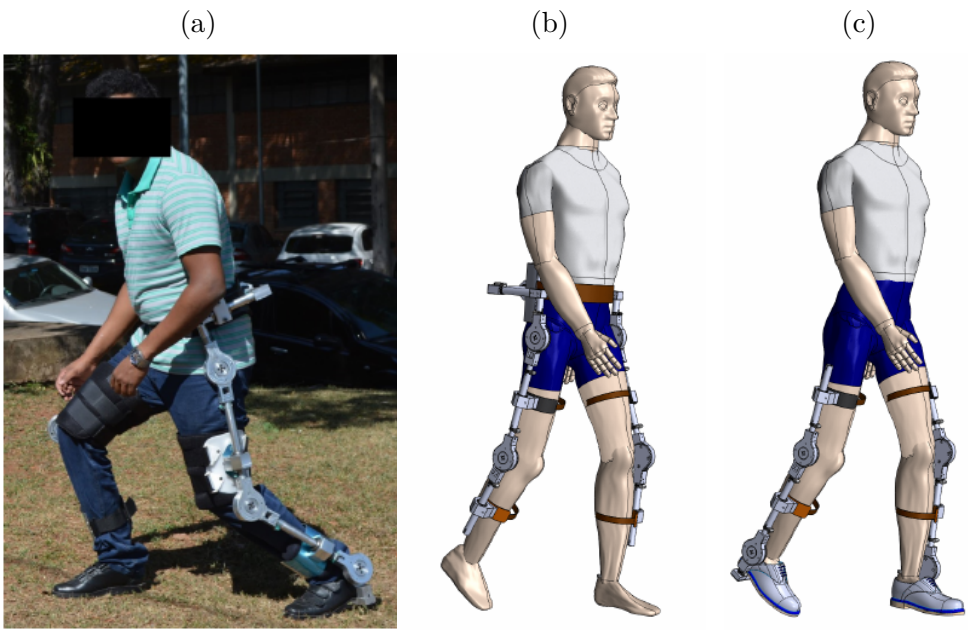
3.1 Rehabilitation Hardware

This section describes the robotic devices and control infrastructure employed for tests during the platform development.

3.1.1 *Harmonic Drive SEAs*

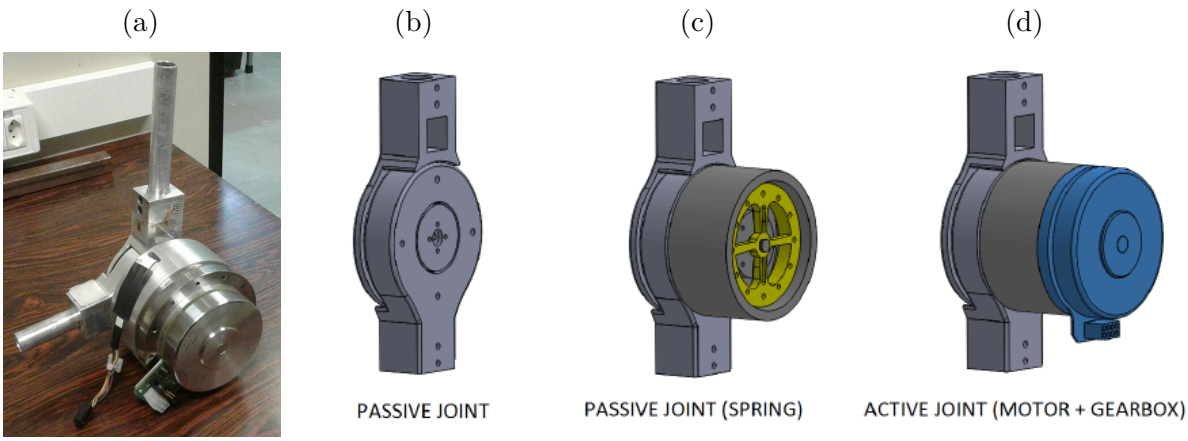
As part of the construction of a new modular exoskeleton for lower limb rehabilitation, *EXO-TAO* (Fig. 19), at the *Robotic Rehabilitation Laboratory* (ReRob) of São Carlos School of Engineering, two SEAs were built for actuation of the hip joints. Those rotary actuators (Fig. 20) are the combination of a compact *Maxon Motor*'s brushless *direct current* (DC) motor ($48V$ of nominal voltage), a high reduction (101:1) and low backlash *Harmonic Drive*® gearbox, a custom torsional spring as the elastic element ($\approx 266N \cdot m/rad$), and an output rotary joint (SANTOS et al., 2017).

Figure 19 – *EXO-TAO* modular exoskeleton (a) full setup in use and (b,c) some of its possible configurations.



Source: Modified from Santos et al. (2017)

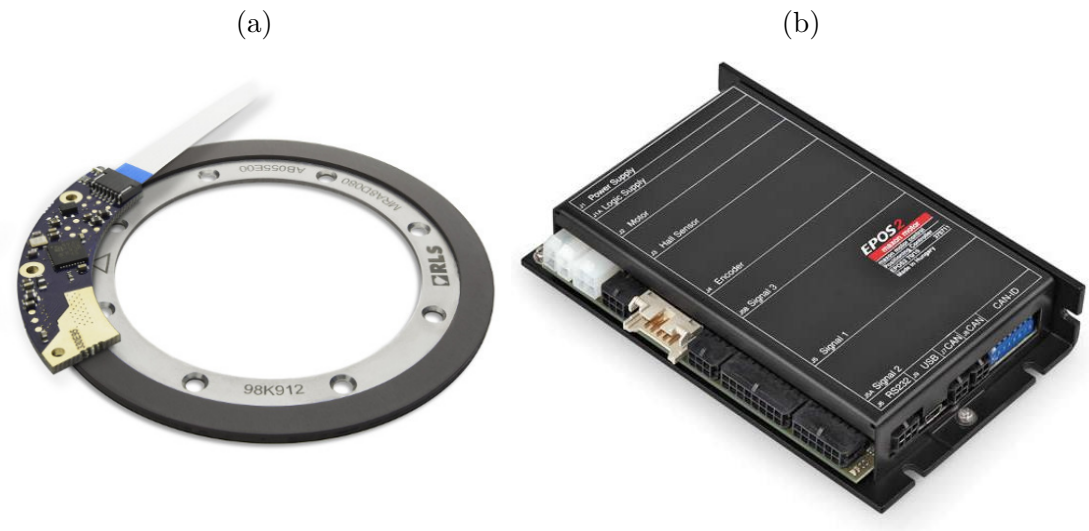
Figure 20 – *EXO-TAO*'s rotary joint (a) assembled with motor and Harmonic Drive gearbox and (b,c,d) some of its mounting options.



Source: (a) The author (b,c,d) modified from Santos et al. (2017)

The measurment of output position – for calculating spring deformation – is obtained with a *AksIM*^{TM} rotary absolute encoder (Fig. 21a), accessible via *Serial Peripheral Interface*[1] (SPI). DC motor encoder (input) position, *Hall* sensor velocity and circuit current are set and read via its *EPOS2* (Fig. 21b) power drive unit, which offers RS232, *CANOpen*[2] and USB communication. Those sensing and actuation features enable implementation of at least basic impedance controllers.

Figure 21 – SEA's sensoring and actuation components: (a) *AksIM*^{TM} rotary absolute encoder and (b) *EPOS2* (Easy Positioning) 70/10 motor drive.



Source: (a) https://www.rls.com (b) https://www.maxongroup.com

[1] http://www.ee.nmt.edu/~teare/ee308l/datasheets/S12SPIV3.pdf [web archive]
[2] http://www.can-cia.org/can-knowledge/

3.1.2 *Real-time Controller*

Being an experimental platform, it is not imperative to make it completely portable, which can be left for future iteractions. That is a reason why there was no restriction on the use of large computers in past developments (CONSONI; SIQUEIRA; KREBS, 2017; CONSONI, 2017), as long as they posed no implementation bottleneck. However, as miniaturized embedded systems evolve, making accessible solutions more feasible, it is worthwhile to check their current applicability to the proposed system.

Also, typical research-focused commercial hardware, although more robust, present significant constraints regarding external hardware and third-party libraries support, which is prohibitive for usage schemes that deviate from the product's originally intended purpose. Now that real-time performance (meeting of update loop deadlines) is more attainable with open general-purpose tools (ALTENBERG, 2016), the compromises of using custom proprietary ones become less justifiable.

That led to the choice of a *Raspberry Pi*[3] (RPi) as embedded computer for the rehabilitation system experimental implementation. With actively maintained support from *Linux* kernels, that platform is a widespread and flexible basis for deploying automation projects, having native USB, *Ethernet* and SPI interfaces, as well as compatibility with virtually any open library. Also, for more recent models, its multi-core ARM (*Advanced RISC Machine*) processor enables effective parallel execution of control and networking threads.

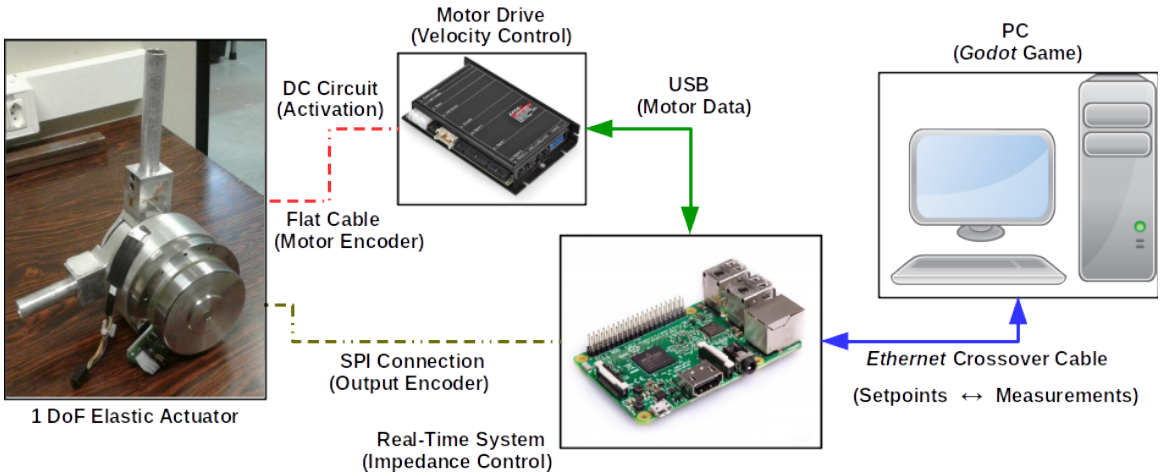
The communication scheme in Fig. 22 shows how the RPi controller integrates with the orthosis actuator: connection with *EPOS2* motor drive is established through USB interface (faster than RS232 or CANOpen alternatives) using the *EPOS* Command Library[4], which is closed source but available as binary for ARM *Linux*; the external *AksIM*^{TM} encoder uses one of the 3 available SPI interfaces. Client interfaces, not limited to the video-game, running on a desktop PC may communicate with that arrangement via either *Ethernet* or *Wi-Fi* LAN.

In order to have a proportionally great processing power, as currently available, for the RPi form factor, a 3B[5] model (1.2 GHz quad-core 64-bit ARM Cortex-A53 CPU, 1GB of RAM) is employed. For enabling full preemptive capabilities to the running operating system, *PREEMPT-RT*[6] kernel patches are applied, although this does not guarantee real-time performance for conventional device drivers.

---

[3] https://raspberrypi.org
[4] https://www.maxonmotor.com/medias/sys_master/8815100788766/EPOS-Command-Library-En.pdf
[5] https://www.raspberrypi.org/products/raspberry-pi-3-model-b/
[6] https://rt.wiki.kernel.org/index.php/Main_Page

Figure 22 – Communication scheme in control of the knee orthosis actuator.



Source: The author, https://www.maxongroup.com and https://www.raspberrypi.org

3.2 Software

This section describes the overall rehabilitation software architecture and the specifics of its individual functional parts.

3.2.1 *Distributed Processing Architecture*

The platform used is a combination of software componentes, running on separate hardware pieces, and providing different functionality to the whole system, like robotic actuators control, *graphical user interfaces* (GUIs), interactive virtual reality, among others. That modularity and distributed operation led to a client-server design, capable of integrating a variable number of collaborative processes, depending on the use case. Fig. 23 shows the example setup of a bilateral telerehabilitation game, with all expected communicating subsystems, to be explained in more detail below.

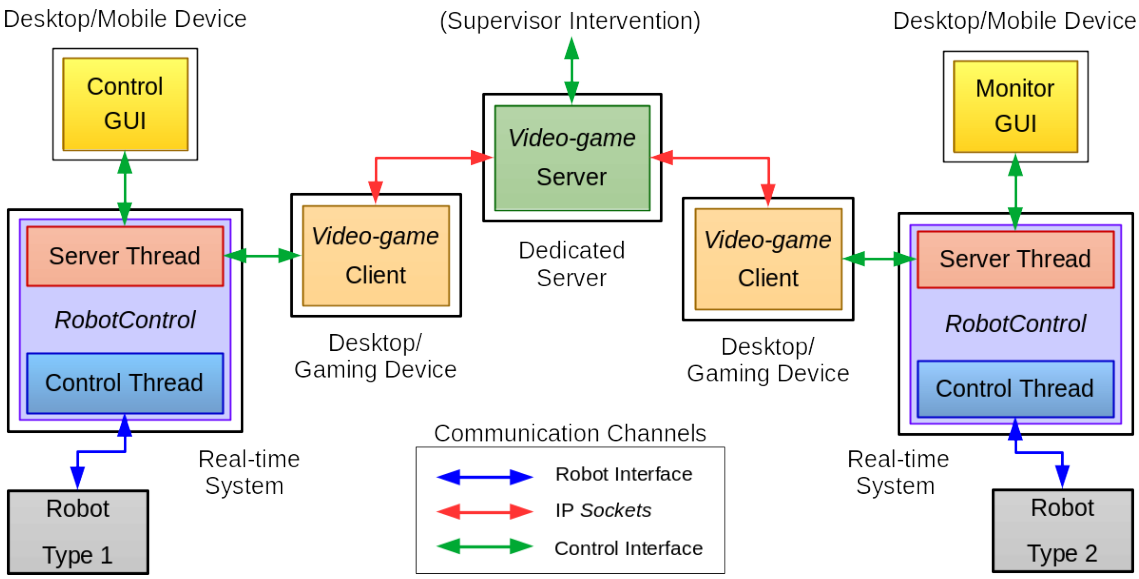
3.2.2 *Inter-process Communication*

Although controller-client communications could be made through faster channels like USB, for practical purposes, a single IPC method was used whenever possible for local and remote (inter-machine) data transfers: IP-based connection over loopback or *Ethernet*, supported by the majority of operating systems (OS) and programming languages through the *BSD sockets* API (*application programming interfaces*).

3.2.3 *Real-time Control Application*

The actual robot control process, named *RobotControl*, is written in *C* programming language (*ISO C99* standard), due to performance requirements and compatibility with libraries and APIs employed. To avoid blocking by other operations and ensure correct

Figure 23 – Diagram of distributed processing example (multiplayer game).



Source: The author, modified from Consoni, Siqueira, and Krebs (2017)

timing (5 ms per step, customizable), the device operation loop is executed in its own parallel thread, which also takes advantage of multi-core CPUs (*computer processing units*), leaving IPC with e.g. monitor GUIs and video-game instances to the main one.

3.2.3.1 Multi-layer configuration structure

As there are many different mechanical geometries, hardware interfaces and algorithms that could be used for rehabilitation robots, the usual approach would involve rewriting specific software to handle each combination. However, in order to effectively carry out the intended objective of making a universal platform, enabling code reuse, *RobotControl* splits generic core functionality from specific implementation details. By means of a hierarchy of configuration files and module system, detailed in Appendix B, the application allows addition of new functionality without modification of its internals.

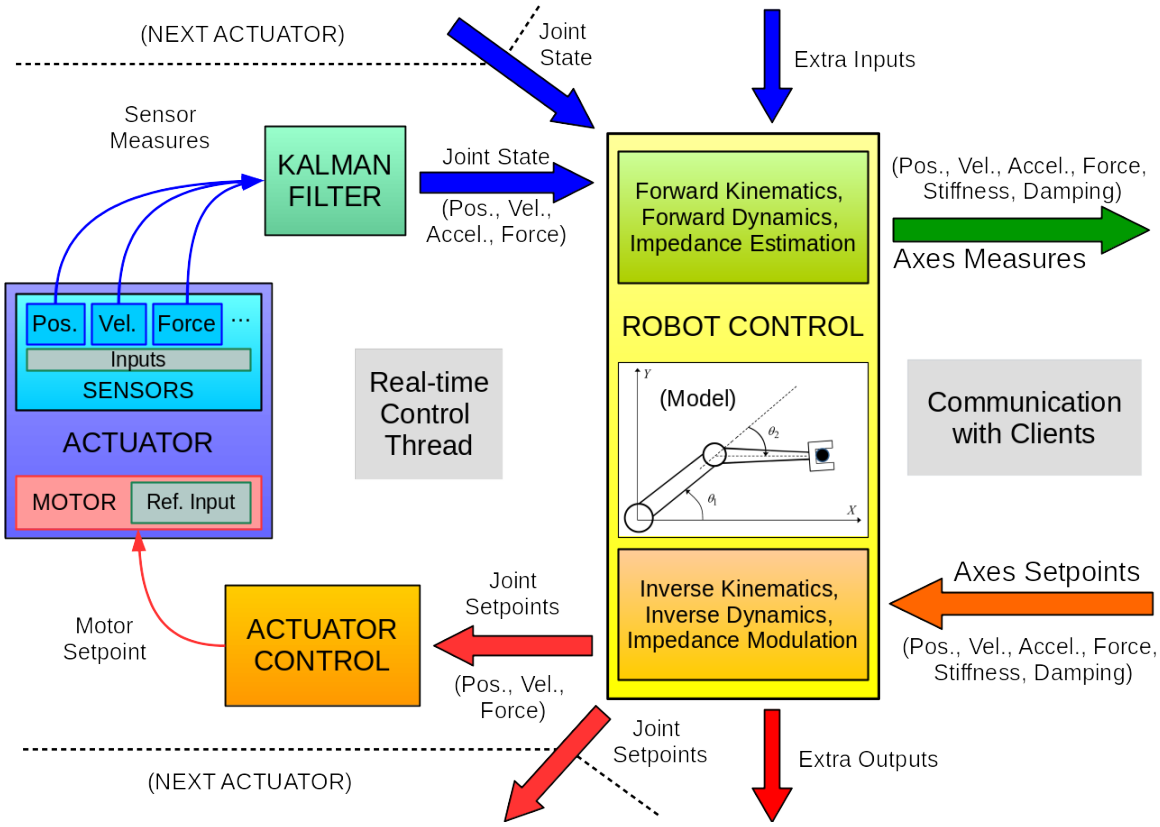
Parameters may be defined by the user at layers of: robot (controller implementation), actuator (measured and generated signals) and sensor/motor (hardware-specifc logic and signal processing options). Higher ones link to lower ones, building a configuration tree, parsed by *RobotControl* during startup. The complete list of possible settings for all levels of device description is presented in Appendix D.

For procedures whose variation between implementations would make a parametrized model unpractical, like data I/O (*input and output*) boards operation and actuator control equations, *RobotControl* allows to write specialized code without violating the generalization approach. By building binary libraries that export defined plug-in APIs, detailed in Appendix C, a developer can setup modules that are loaded and called by the core executable at runtime, optimizing behaviour for a given task.

3.2.3.2 Control Data Flow

During each processing step of the control loop, real-time actuation data pass through a series of transformations from setpoint acquisition to motor output, as illustrated by Fig. 24. For each joint actuator of a robot, a *Kalman* filter makes possible to combine signals from redundant sensors or estimate values for missing measurements, keeping always the same set of state variables. The right side arrows represent remote measurement and setpoint transfers performed through the *RobotControl* proxy server.

Figure 24 – Diagram of data flux and transformations inside *RobotControl*.

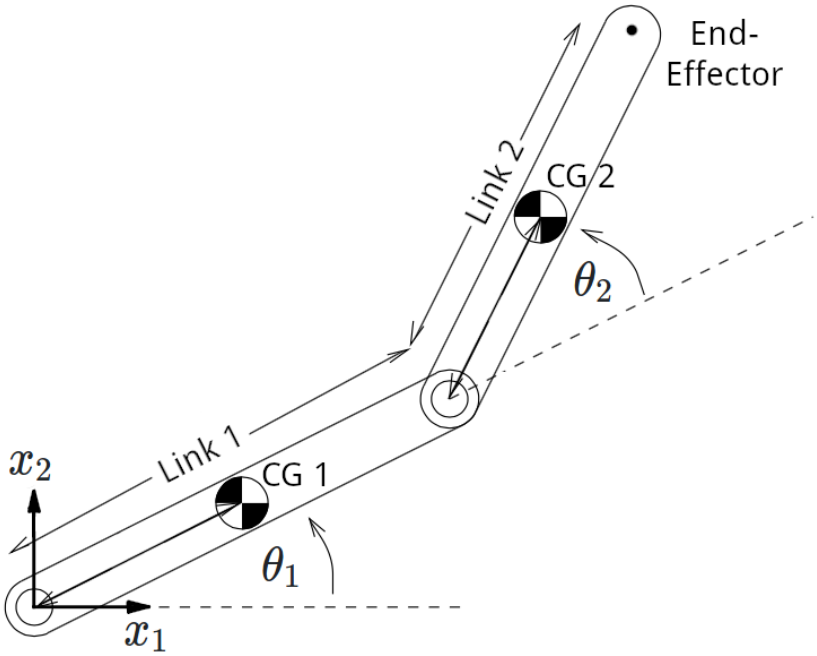


Source: The Author

From the point of view of a *RobotControl* client application, operating the robot or just visualizing its data, kinectic variables of interest do not necessarily match the *degrees of freedom* (DoFs) over which actuators have direct control. Fig. 25 gives the example of a 2-link robotic arm where the cartesian position (x_1, x_2) of an end-effector tool could only be controlled via motors and sensors on its revolution joints (coordinates θ_1, θ_2).

Making the required conversions from one coordinate system to another would demand from graphical interface developers knowledge about particular robot kinematics and dynamics, violating intended device abstraction. Besides, that would increase complexity of a frontend implementation with routines not related to user interaction. Thus, the approach taken requires that such task is performed on the real-time controller, as it already runs robot-specific code, avoiding duplication of effort.

Figure 25 – Example of robotic manipulator with distinct actuator and effector coordinates.



Source: Modified from https://engineering.stackexchange.com

For clients, then, the set of variables accessible on each device's DoF is the result of internal coordinate transformations over original joint state, here called "axis" values – a reference to joysticks' analog axes. Custom control modules update logic is responsible for performing conversions from joint to axis measurements, and from axis to joint target setpoints. The process also involves joints stiffness, damping and inertia estimation and impedance modulation.

In order to keep consistency in data transformations across third-party developed plug-ins and allow interoperation between them, the convention in Tab. 2 is adopted for measurement (output) and setpoint (input) variables.

Table 2 – Control variables units convention

Control Variabe	**Translational Unit**	**Rotational Unit**
Position	meter (m)	radian (rad)
Velocity	meter / second (m/s)	radian / second (rad/s)
Force/Torque	Newton (N)	Newton · meter ($N \cdot m$)
Acceleration	meter / second^2 (m/s^2)	radian / second^2 (rad/s^2)
Stiffness	Newton / meter (N/m)	Newton · meter / radian ($N \cdot m/rad$)
Damping	$N \cdot s/m$	$N \cdot m \cdot s/rad$
Inertia	kg ($N \cdot s^2/m$)	$kg \cdot m^2$ ($N \cdot m \cdot s^2/rad$)

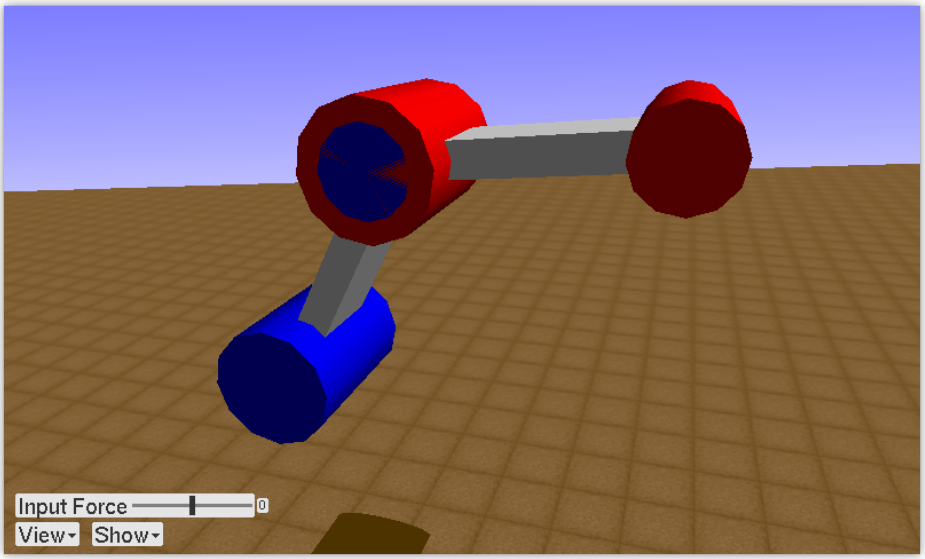
Source: The author

3.2.4 *Human-machine Mechanical Modeling*

Depending on the complexity of orthosis, exoskeleton or end-effector coupling with the human user, writing down the algorithms for their coordinate transformations or forward/inverse kinematics and dynamics manually could be a burdensome task. In order to cut implementation time and prevent mismodeling errors, being able to automate or generalize parts of that process is desirable.

To meet that need, Consoni, Peña, and Siqueira (2018) and Peña et al. (2019) use *OpenSim*[7], a freely distributed (under the *Apache 2.0*[8] license) software for biomechanical simulation (Fig. 26). It ships with a set of generic human skeleton and muscle models that could be adjusted to specific individuals through an interactive application, generating a properly scaled version from supplied subject's anthropometric data (SETH et al., 2011; VIVIAN; REGGIANI; SARTORI, 2013).

Figure 26 – *OpenSim* simulation of a 2-link manipulator model.



Source: The author

Both original and modified system descriptions are stored as XML files and can be loaded by *OpenSim* libraries, available for programs written in *C++*, *Java*, *Python* or *Matlab*. The musculoskeletal model resulted from the static optimization could then be supplied with sensors data to calculate patient effort, as the software also provides articulation torques estimation from input reaction forces (inverse dynamics) or sEMG signals applied to muscles (forward dynamics) (SETH et al., 2011; SARTORI; FARINA, 2016; PEÑA et al., 2019).

[7] http://opensim.stanford.edu/
[8] http://www.apache.org/licenses/LICENSE-2.0

As a setback, although modifications to the open-source code could add support for online inverse kinematics (already being worked on for current release release[9]), preview results from real-time *OpenSim* processing, presented in (PIZZOLATO et al., 2016), show how the computational cost of that approach scales intensely with model-complexity, to the point that iteration times go much above the 5 milliseconds limit used in *RobotControl*'s update loop. As an alternative, Consoni, Peña, and Siqueira (2018) and Peña et al. (2019) propose using offline simulation results from a scaled and reliable model to pre-train a neural network, whose runtime processing should be much cheaper.

Another potentially more efficient but less straightforward solution would be to use code generators such as *RobCoGen*[10] (FRIGERIO et al., 2016) or *OpenModelica*[11] (DEBRAY, 2015; FRITZSON; THIELE, 2016), which produce compilable code implementations of dynamic equations from text- or graphic-based descriptions of the target system, like the DC motor in Fig. 28. At the cost of maintaining one library for each model, far more optimized algorithms could be employed.

Figure 27 – Same *Modelica* model represented in (a) source code and (b) block diagram.

(a)

```
model DCMotor
   Resistor R(R=100);
   Inductor L(L=100);
   VsourceDC DC(f=10);
   Ground G;
   ElectroMechanicalElement EM(k=10,J=10, b=2);
   Inertia load;
equation
   connect(DC.p,R.n);
   connect(R.p,L.n);
   connect(L.p, EM.n);
   connect(EM.p, DC.n);
   connect(DC.n,G.p);
   connect(EM.flange,load.flange);
end DCMotor
```

(b)

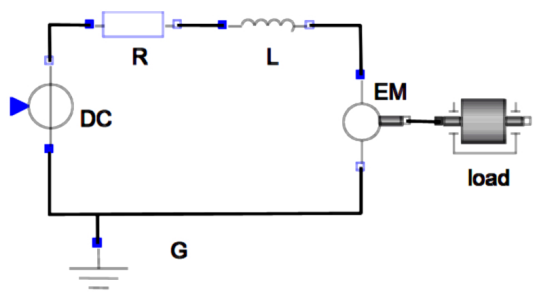


Figure 28 – Source: Modified from Fritzson and Thiele (2016)

As long as the written or generated code has compiler or binary compatibility with the *C* programming language, it is able to work as a *RobotControl* plug-in by exporting the standardized *C* interface (Appendix C.2).

3.2.4.1 Network proxy server

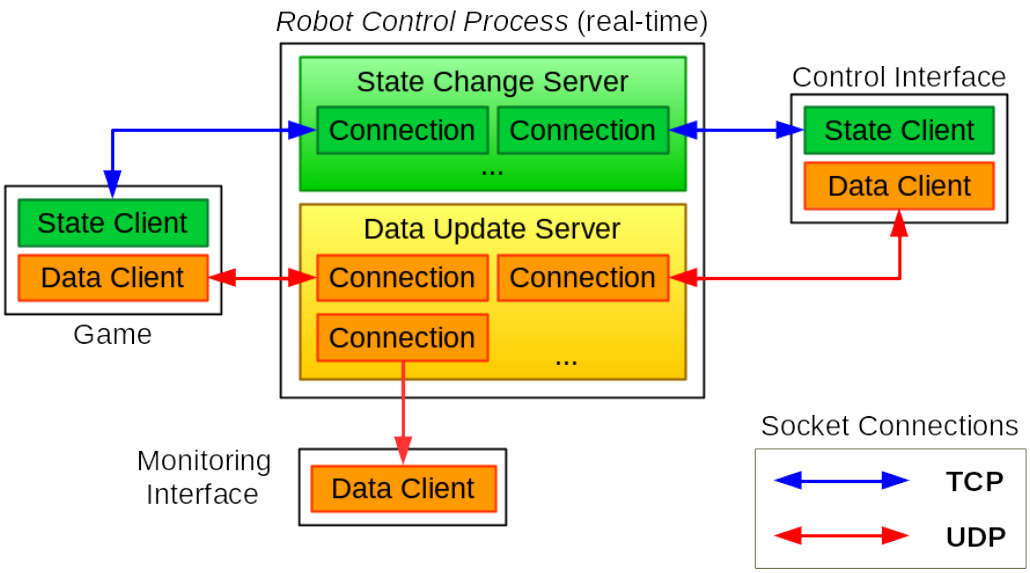
Running on its main real-time thread (20 ms update interval), *RobotControl* network server component acts as a proxy between control logic and client applications. From remote peers, it takes requests and sends back information through IP sockets, using I/O

[9] https://github.com/RealTimeBiomechanics/rtosim
[10] https://robcogenteam.bitbucket.io/index.html
[11] https://openmodelica.org/

polling to handle multiple connections. Internally, it shares exchanged values with the independent control thread, the effective data producer/consumer, via synchronized calls.

Fig. 29 shows how there are actually two server instances, represented by large colored boxes, handling their own specific sets of asynchronous connections with remote client sockets. Black borders delimit processes running on different machines, small colored boxes represent open communication endpoints.

Figure 29 – Diagram of *RobotControl* servers and its possible types of client connections.



Source: The author, modified from Consoni (2017)

Depending on the type of data transmitted through IP communication channels, different underlying protocols are used, considering their intrinsic characteristics. TCP, slower and more reliable, is used for critical information sent only a few times, like initialization and state change commands; UDP, faster and more error prone, works for continuous updates, when lower latency matters the most and the loss of a message would be quickly compensated by a following one.

3.2.5 *Robot Control Messaging Format*

Client applications communicating with *RobotControl* through IP connections shall use port 50000 (customizable) and fixed-size 512 bytes messages, as data packets below 1 kilobyte are very likely to be transmitted without fragmentation (HALL, 2011). Depending on the type of message sent or received, a specific data format is defined.

Messages requesting state changes or robot information are sent by clients to server over TCP sockets, as a single byte containing one of the opcode values listed in Appendix A.1. On success, a reply is sent back with the same code. Depending on the opcode, it is followed by additional data.

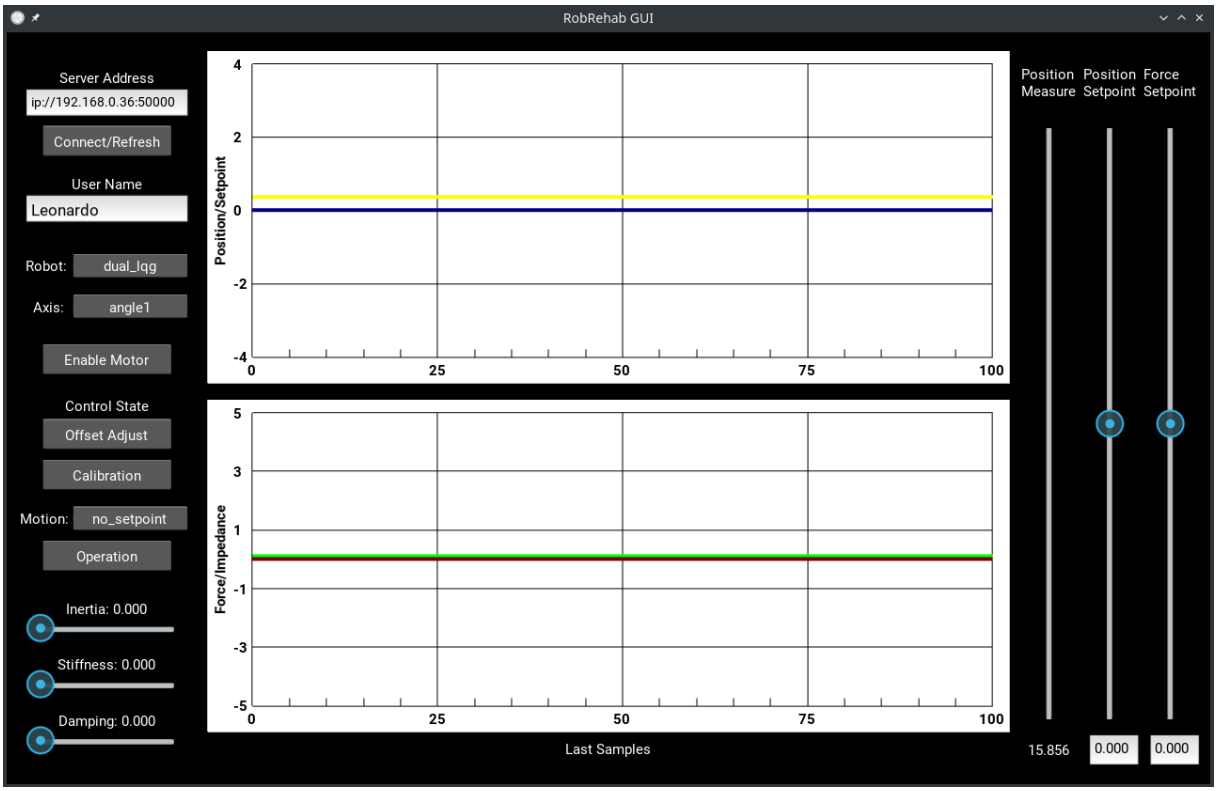
On the other hand, axis measurements and setpoints are transported by UDP messages. Arrays of DoF indexes and related single precision (4 bytes) floating-point values are serialized to and deserialized from these network packets following a predefined data format, as detailed in Appendix A.2.

3.2.6 *Control and Visualization GUIs*

Basic user interfaces for remote robot control or data visualization may be created with some specialized toolkit, like *TK*, *Qt* or *Kivy*, in compliance with defined IPC channels setup (Fig. 29) and message structure (Appendix A).

The implementation in Fig. 30 provides access to most of *RobotControl*'s features: buttons on the left side issue request callbacks (Appendix A.1); central graphs and right sliders show data received from update messages (Appendix A.2).

Figure 30 – Control GUI created with the *Kivy* framework for *Python*.



Source: The author

3.2.7 *Virtual Telerehabilitation Games*

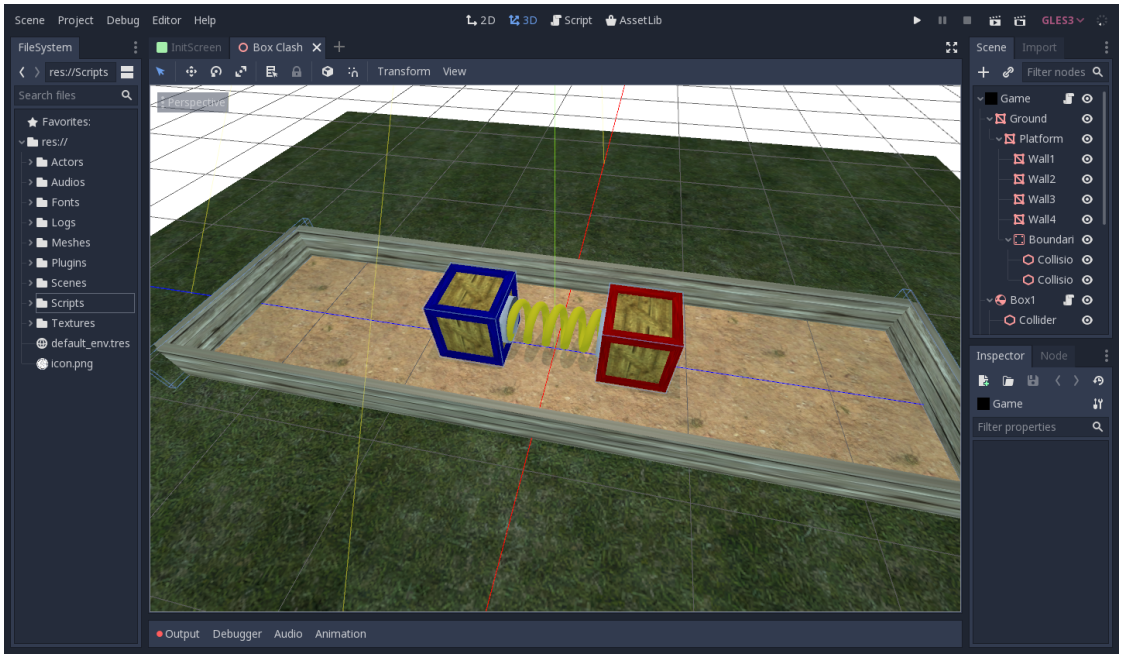
One of the essential parts of the telerehabilitation platform is the video-game through which users interact. In previous works (CONSONI; PASQUAL, et al., 2016; CONSONI; SIQUEIRA; KREBS, 2017), a common and reusable framework was developed first, and then used to create several games with different purposes for testing.

3.2.7.1 Development tool

When looking for a game development solution, game engines were prioritized, as they already provide a variety of working software components for graphics, sound, scripting, input handling, GUIs and networking, as well as multi-platform support (including mobile). Ease of use, community support and availability of learning material for implementation of different functionality was also considered.

Due to the heavy system resource requirements and license restrictions of proprietary tools such as *Unity3D* and *Unreal*, it was decided to adopt a open source alternative: *Godot*[12] engine, a MIT[13] licensed game creation software which is actively developed and has a growing user base. Even being considerably lightweight, its editor (Fig. 31) has all the features needed for teleoperated therapy games and a custom *Python*-like programming language (*GDScript*).

Figure 31 – Graphical development environment of *Godot* editor



Source: The author

3.2.7.2 Multiplayer game framework

For the common multiplayer game framework – a set of base *GDScript* classes (Appendix E) and *Godot* scenes (editor files) – a master server scheme was employed. That server is a special instance of the game implementation with unique behaviour, promoting authoritative or semi-authoritative synchronization across connected client executables, with whom players interact using the rehabilitation robots.

12 https://godotengine.org/
13 https://tldrlegal.com/license/mit-license

3.2.7.3 Network synchronization

Instead of dealing with IP sockets directly, in *Godot* multiplayer synchronization across server and client nodes is performed via *remote procedure calls*[14] (RPC), a UDP-based callback subsystem which offers *quality of service* (QoS) options – regarding data integrity and ordering reliability – while saving the coder from low-level manipulation of network buffers.

Godot's RPC mechanism also seems do deal better with network congestion issues, experienced with *Unity3D* (CONSONI; PASQUAL, et al., 2016), so that custom traffic optimization strategies previously attempted could be dropped. That way, game state sync calls are always tied to physics engine updates, happening every 20 milliseconds.

3.2.7.4 Input calibration and scaling

Upon loading games based on this framework, the configuration interface in Fig. 32 is presented to players. It allows to set the player name for data logging, define device's interface type (e.g. joystick or keyboard conventional input or LAN IP for *RobotControl*'s clients) and string ID required for connection, select from available device configurations and controllable axes/DoFs (whose position and force values are displayed in the central sliders), toggle device states (enable/disable, offset, calibration and operation), choosing the game scene to be played and binding its variables to particular device's axes.

Figure 32 – Configuration and calibration screen for rehabilitation games.

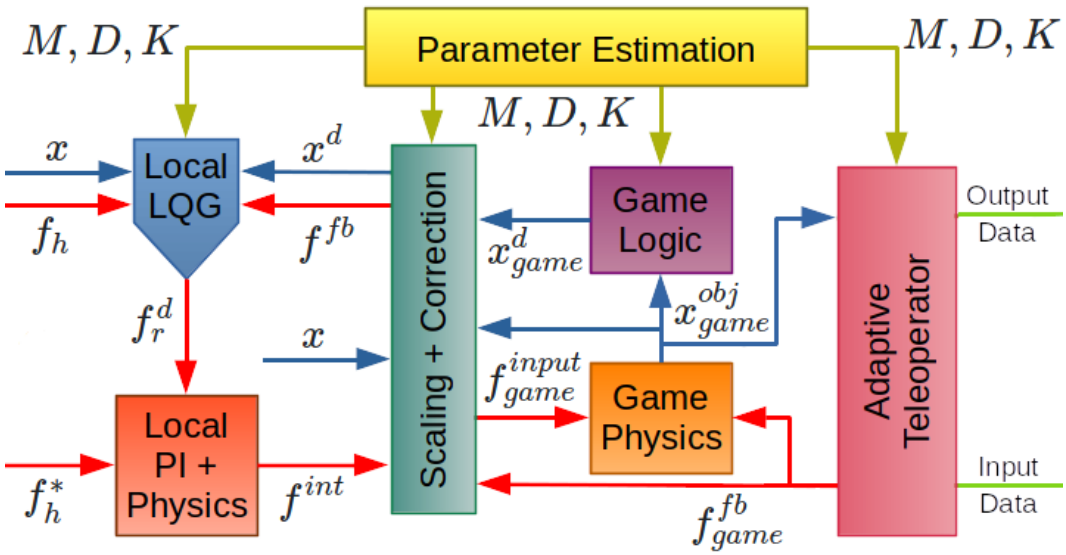
Source: The author

[14] https://godot.readthedocs.io/en/3.0/tutorials/networking/high_level_multiplayer.html

The offline calibration phase (T_{cal} duration) is intended for a robot operating in fully compliant mode, in order to take measurements for initial offset x^{off} and maximum range Δx – from lower to upper bounds – of patient measured movement on each DoF. Those values are then applied for online bidirectional scaling of local positions (x, x^d) and forces (f^{int}, f^{fb}), represented in Fig. 33, so that the player-controlled object/avatar is mostly kept inside the expected virtual working area:

$$\begin{gathered} x^{player}_{game} = \frac{(x - x^{off})}{\Delta x}\alpha^x, \quad x^d = \frac{x^d_{game}\Delta x}{\alpha^x} + x^{off}, \quad \text{where} \\ x^{off} = x(0), \quad \Delta x = x^{max} - x^{min}, \quad x^{max} = \sup_{t\in[0,T_{cal}]} x(t), \quad x^{min} = \inf_{t\in[0,T_{cal}]} x(t). \end{gathered} \tag{3.1}$$

Figure 33 – Diagram of axis scaling added to multiplayer game terminal.



Source: The author

Following initial set-up, the gameplay screen is displayed, and the activity is started after all players are connected to server. Estimated active stiffness $\hat{K}_h$ and damping $\hat{D}_h$ are used as correction gains for player force input f^{input}_{game}, in order to track scaled effector position x^{player}_{game}. The manipulated virtual object, at position x^{obj}_{game} and velocity $\dot{x}^{obj}_{game}$, is left as a pure inertia M_h (the more constant dimension) for game physics processing:

$$\begin{gathered} f^{input}_{game} = (f^{int}_{game} + \hat{K}_h(x^{player}_{game} - x^{obj}_{game}) - \hat{D}_h\dot{x}^{obj}_{game})\alpha^f, \quad \text{where} \\ f^{int}_{game} = (f^{int} - f^{off})\alpha^x/\Delta x, \quad f^{off} = f^{int}(0), \quad f^{fb} = f^{fb}_{game}\Delta x/\alpha^x. \end{gathered} \tag{3.2}$$

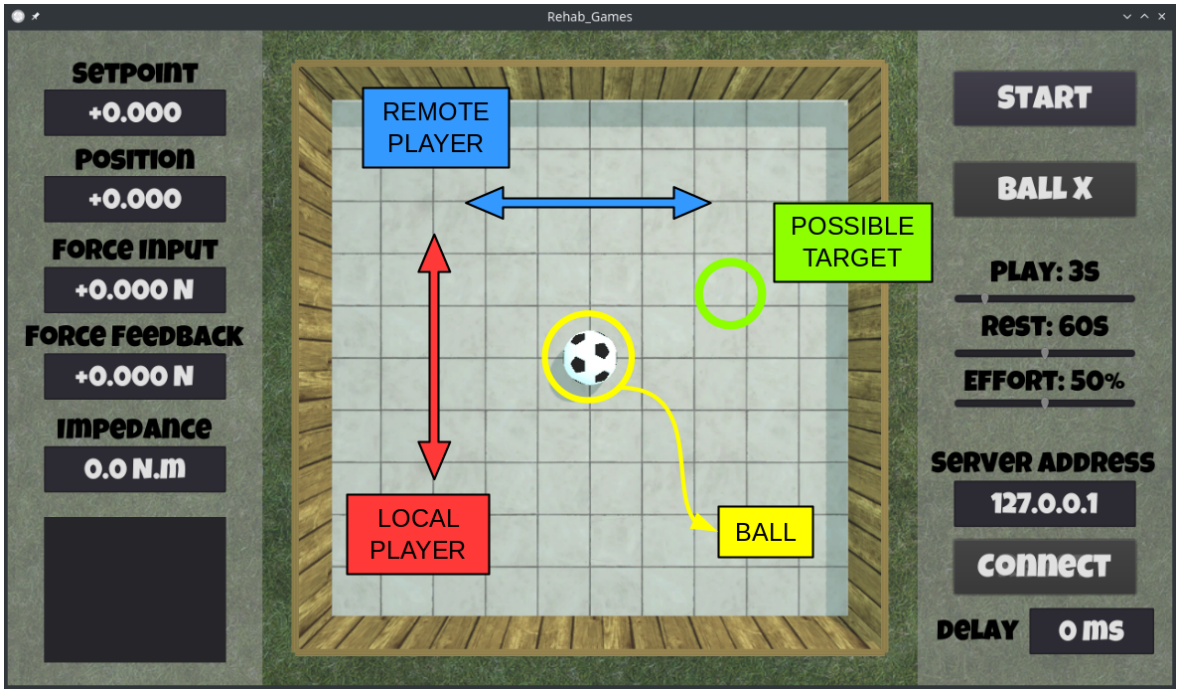
Factors α_x and α_f work as free parameters for, respectively, specific game space movement dimensioning and sizing individual players force input for difficulty balancing.

3.2.7.5 Interactive gameplay

Following Jarrassé, Charalambous, and Burdet (2012) definitions in (Section 2.4.3), 2 video-games were developed to cover most of the interactive multiplayer task cases:

- *Move Ball* (Fig. 34): A basic 2-axis virtual activity set for players to cooperate (agonistic distinct roles) in moving a object to defined postions in a plane. Each player-controlled axis moves the ball in a different direction inside the central board. If the targets for each direction are detached, the game is turned into a co-active (divisible subtasks) task. Remote players could even share control over the same axis, enabling basic competition/collaboration. Camera rotation makes the client's moved axis always seem as vertical.

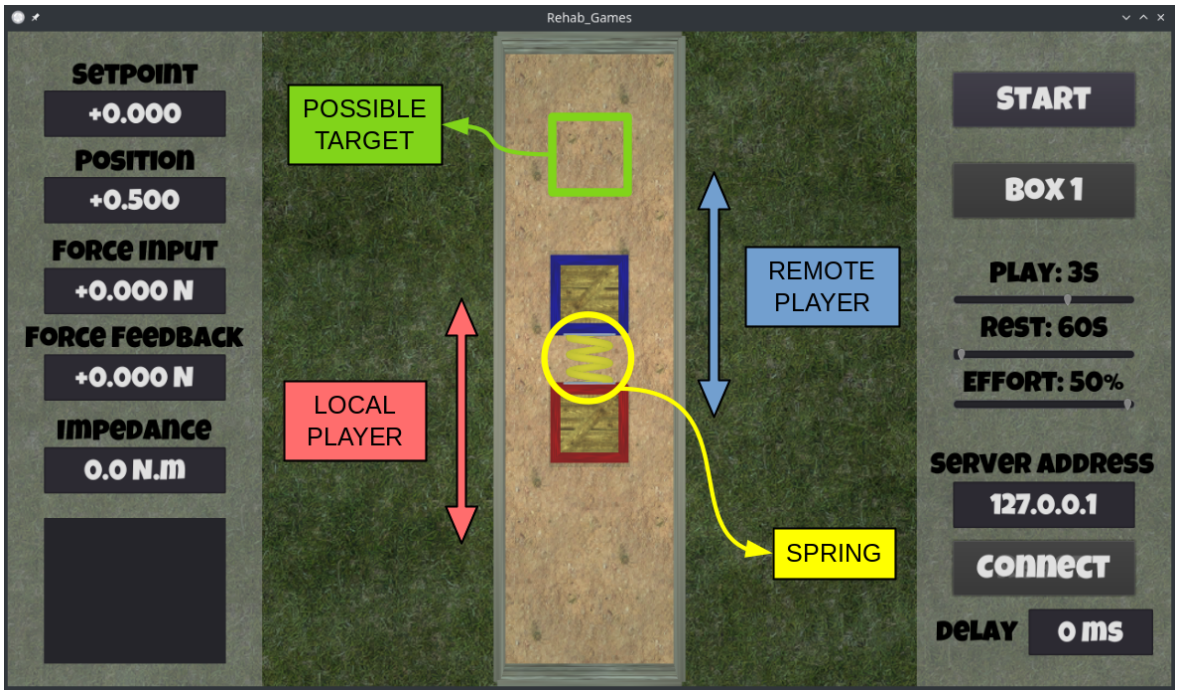
Figure 34 – *Move Ball* gameplay screen elements.



Source: The author

- *Box Clash* (Fig. 35): Originally created in Consoni (2017), it was adapted to evaluate performance in agonistic and antagonistic tasks. It has a simplified layout, where two player-controlled boxes continuously exchange forces through a spring element that connects them. Local players consistently view their input affecting the bottom box. During play time a transparent green box marks the target position for each player. As the spring allows the boxes to push or pull each other, this scene could be used for either competitive (different targets) or collaborative (same target) gameplay.

Figure 35 – *Box Clash* game screen elements.



Source: The author

3.3 Control Algorithms

After a description of hardware and software infrastructure, here the actual control strategies implemented on top of it are discussed. The algorithms proposed and tested are implemented as replaceable modules to the game framework.

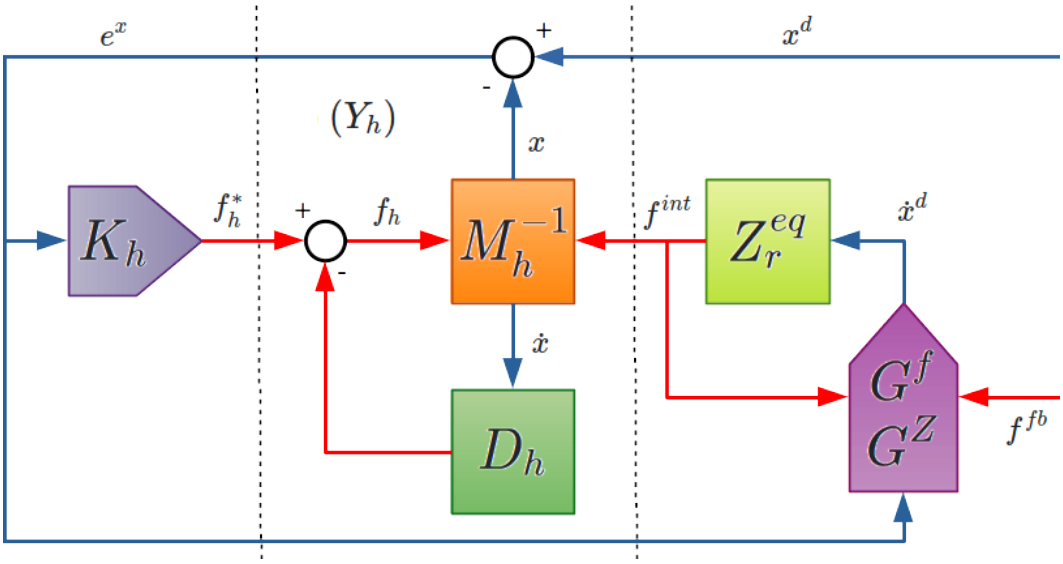
3.3.1 *System Linearization*

Considering how the SEA's elastic element decouples actuator and body dynamics to some extent, they are treated as an impedance-admittance pair. Fig. 36 shows how the interaction force f^{int} is seen as output from robot's Z^{eq} and input for human Y_h transfer function. Similarly, impedance control action f_r^d from G^Z is used for calculation of desired motor velocity setpoint $\dot{x}^d$ by force regulator G^f (Eq. 2.2).

While internal G^f gains are manually adjusted, the higher-level local model parameters are generated via least squares method (Eq. 2.11), for minimum variance (quadratic deviation), on a window of $n = 200$ samples (1 second). Here, real-time estimates of human dynamics variables K_h, D_h and M_h are obtained as:

$$f^{int} = \begin{bmatrix} -e^x \\ \dot{x} \\ \ddot{x} \end{bmatrix}^T \begin{bmatrix} K_h \\ D_h \\ M_h \end{bmatrix} \implies \begin{bmatrix} \hat{K}_h \\ \hat{D}_h \\ \hat{M}_h \end{bmatrix} = (X^T X)^{-1} X^T \begin{bmatrix} f_1^{int} \\ \vdots \\ f_n^{int} \end{bmatrix}, \quad X = \begin{bmatrix} -e_1^x & \dot{x}_1 & \ddot{x}_1 \\ \vdots & \vdots & \vdots \\ -e_n^x & \dot{x}_n & \ddot{x}_n \end{bmatrix}. \quad (3.3)$$

Figure 36 – Control diagram of unified human and robot dynamics.



Source: The author

Unlike 2.7, only $\hat{M}_h$ is then used to compose the time-varying matrices for state-space representation of local subsystem (meaning $M = \hat{M}_h, D = 0, K = 0$), while $\hat{K}_h$ and $\hat{D}_h$ are used to correct human game input (Eq. 3.2).

In order to prevent unsafe usage of results from ill-conditioned matrix equations, a preliminary calibration phase, keeping attached patient with relaxed limbs ($f_h^* \approx 0, K_h \approx 0$), is performed in order to establish the secure lower bound of admissible values, as done in Miyoshi, Terasima, and Buss (2006) and Atashzar, Shahbazi, et al. (2017).

3.3.2 *Optimal AAN Force Input*

Regardless of the force reflection strategy used, local assistive/resistive input $f^{a/r}$ (shown in Fig. 33), applied for task reference x^d tracking, are also generated using an iterative form of LQG compensation and combined with feedback f^{fb}:

$$\begin{aligned} z_{k+1} &= A_k z_k + B_k u_k + L_{k+1}(x_k - x_k^d - C(A_k z_k + B_k u_k)), \quad \text{where} \\ u_k &= f_k^{ext} + f_k^{fb}, \quad f^{ext} = f_h + f^{a/r}, \quad f^{a/r} = -G^Z z, \quad f_r^d = f^{a/r} + f^{fb}. \end{aligned} \tag{3.4}$$

Instead of conflating human effort f_h and remote intervention f^{fb} with any external disturbance, they are explicitly added with AAN control action $f^{a/r}$ to resulting input u, provinding some foreknowledge on their contribution to motor task fullfilment. LQG regulator G^Z and *Kalman* gain L are updated every time step from estimates $\hat{M}_h$ and Δt.

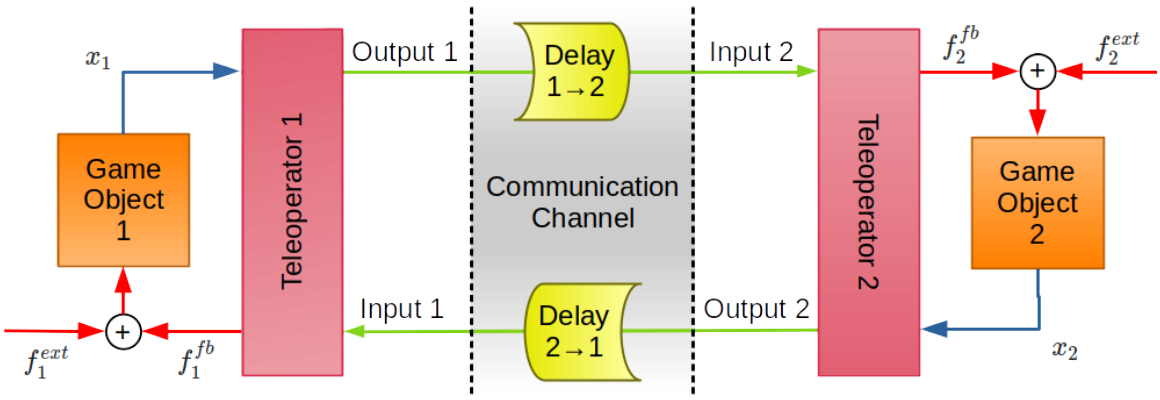
As f_h and f^{fb} work in favor of trajectory following, the robot intervention f_r (setpoint f_r^d) assistance component $f^{a/r}$ tends to decrease in magnitude. The reduced error leaves room for adjusting the cost function for increased robustness ($\rho \to 0$).

3.3.3 *Teleoperation strategies*

Sharing implementation between game client and server, teleoperators here were made symmetrical, with input kinematic data – position x, velocity $\dot{x}$ – and output force f^{fb}, inserted (after eventual correction) as external contribution for the local LQG.

Fig. 37 shows that mirrored layout: master and slave terminals do not need to be defined as they are interchangeable. In order to simplify representation, only the virtual part of force reflection is considered, without robot's impedance/position controller and variables scaling (Fig. 33).

Figure 37 – Symmetrical force feedback teleoperator layout.

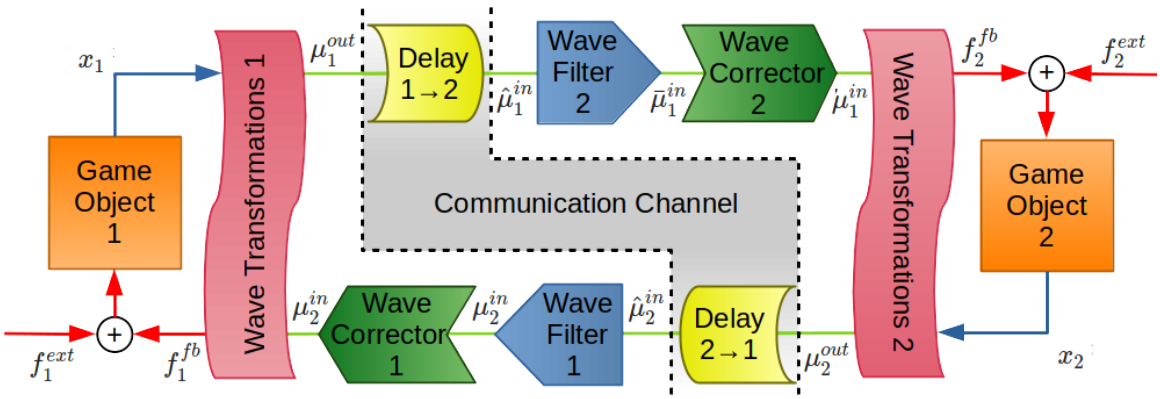


Source: The author

3.3.3.1 Wave variables w/ position tracking

For reference, a more conservative approach based on wave variables (Section 2.3.1.3) is attempted first. Fig. 38 represents the bilateral transmission of signals μ and their integrals U (encoding position data), with low-pass filtering ($\lambda = 0.5$) and adaptable scaling ($\varepsilon = 0$) for delay-independent stable interaction, according to Eqs. 2.13-2.18.

Figure 38 – Diagram of wave variables teleoperator for remote interaction game.



Source: The author

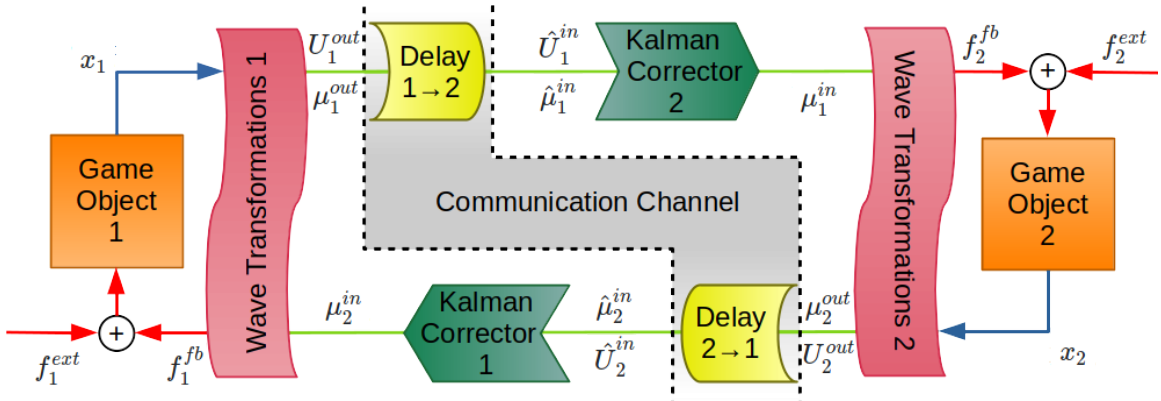
With the intent to achieve online adaptation to remote environment dynamics (K_e, D_e, M_e) – that include not only co-player behaviour but virtual world reactions such as resistance and collisions – locally estimated parameters are also transmitted as setpoint for wave impedance b, continuously recalculated as the average of both sides:

$$b = ((\hat{D}_h + \hat{M}_h) + (\hat{K}_e + \hat{D}_e + \hat{M}_e))/2. \tag{3.5}$$

An alternative strategy, shown in Fig. 39, inspired by Munir and Book (2002) is also attempted to improve transparency: the application of *Kalman* prediction Eq. 2.23 to both μ and U signal pairs:

$$y^\mu = \begin{bmatrix} U^{in} \\ \mu^{in} \end{bmatrix}, \; U^{in} = \int_0^{t-T} \mu^{out} d\theta, \quad F^\mu = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \; H^\mu = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{3.6}$$

Figure 39 – Diagram of wave variables teleoperator for remote interaction game.



Source: The author

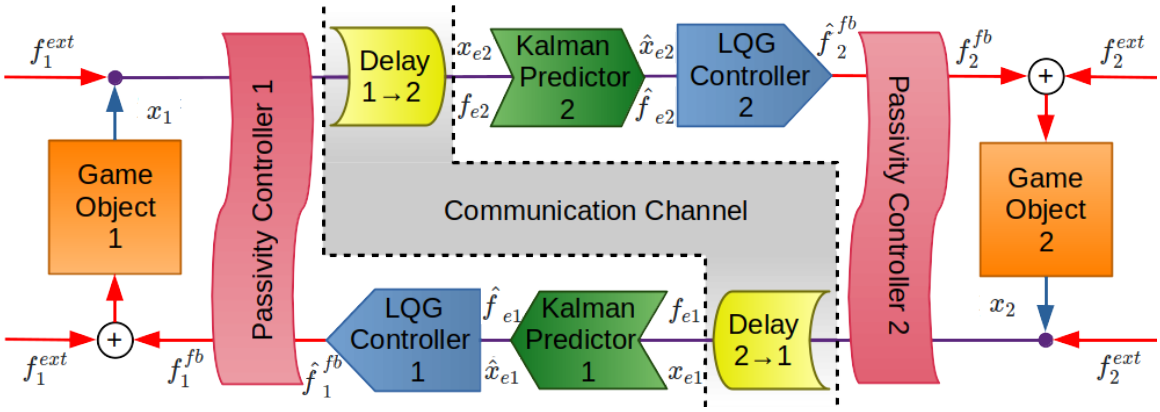
Here, y^μ, F^μ and H^μ replace their counterparts y, F and H in the previosly described predictor equation (Eq. 2.23). The stochastic filter also replaces Eqs. 2.16-2.18, as they're made mostly redundant by the information from wave integrals.

3.3.3.2 LQG w/ direct force feedback

A less restrictive method, named LQG-FFB, is proposed with a focus on transparent and nonpassive interaction. It consists of direct transmission of locally measured f^{ext} (Eq. 3.4) as environmental reaction force f_e. Fig. 40 shows how force and position signals are combined (purple line) to produce the feedback f^{fb}.

Here, a second LQG compensator is applied. Predicted remote position $\hat{x}_e$ and velocity $\hat{\dot{x}}_e$ are taken for correction of synchronization error z^e and estimated reflected force $\hat{f}_e$, using L^e *Kalman* and G^e gains with the same model matrices A and B:

Figure 40 – Diagram of LQG prediction teleoperator for remote interaction game.



Source: The author

$$
\begin{aligned}
z^e_{k+1} &= A_k z^e_k + B_k f^{fb}_k + L^e_{k+1}(x_k - \hat{x}_e k - C(A_k z^e_k + B_k f^{fb}_k)), \\
\text{where} \quad f^{fb}_k &= M - TDPC(\hat{f}^{fb}_k), \quad \hat{f}^{fb}_k = -G^e_k z^e_k + \hat{f}_e k.
\end{aligned}
\tag{3.7}
$$

The forward in time estimates $\hat{x}_e$, $\hat{\dot{x}}_e$ and $\hat{f}_e$, for transparency augmentation, also follows Eq. 2.23, with their corresponding received position and force signal vectors (y^x, y^f), observers (H^x, H^f) and shared prediction matrix $F^{x,f}$:

$$
y^x = \begin{bmatrix} x_e \\ \dot{x}_e \end{bmatrix}, \; H^x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad y^f = \begin{bmatrix} f_e \end{bmatrix}, \; H^f = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}^T, \quad F^{x,f} = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.8}
$$

The initially calculated $\hat{f}^{fb}$ is then corrected on demand by the M-TDPC regulator described in Eq. 2.22 ($\Gamma = 0.7$), where the local damping estimation $\hat{D}_h$ from Eq. 3.3 is applied, in order to produce the actually delivered feedback force f^{fb}.

In more complex scenarios, where the real or virtual environment may significantly interfere with operation, its mechanical properties could also take part in the state space model with modified matrices A' and B', for G^e and L^e:

$$
f_h = \hat{K}_h e^x - \hat{D}_h \dot{x}, \quad A' = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ \frac{-\hat{K}_e}{\hat{M}_h + \hat{M}_e} & \frac{-\hat{D}_e}{\hat{M}_h + \hat{M}_e} & 1 \end{bmatrix}, \; B' = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{\hat{M}_h + \hat{M}_e} \end{bmatrix}. \tag{3.9}
$$

That suggested approach would turn synchronization logic more complex, though, and it is expected that the combination of position control and direct force reflection is enough to deliver a proper perception of collision, motion drag, etc.

3.4 Advantages over Previous Works

Compared to previous solutions found in related literature, the strategies proposed here attempts to improve them in 3 distinct but integrated aspects:

- Online identification of local and remote environment properties and network delay for recurrent correction of teleoperation signals, supplementing static formulations and offline adjustment procedures;
- Model for estimation of user effort – from robot data – during operation time and explicit inclusion of external forces – instead of accounting them as disturbance – in tha AAN algorithm in order to avoid providing unnecessary assistance;
- Forward and backward scale conversions that enable combination of different controllers into the same rehabilitation collective game, while still allowing exercising player's physical performance to impact the activity and be used for difficulty balancing, avoiding the development of applications and activities around specific and hardly extensible setups.

What ties them together is common usage of the linearization procedure from Eq. 3.3 and state-space matrices A and B. That way, a clear interface is established between the 3 subsystems, which are able to function in a mostly independent manner.

3.5 Closing Remarks

Dealing with the issue of robotics-enabled telerehabilitation via multiplayer games over *Internet* is a multidisciplinary effort. The knowledge on available technologies is scattered across many research lines, so that incremental improvements are usually proposed for each separate topic.

What is expected from this unified model concept is to have a more comprehensive perspective on every component requirements and optimization trade-offs for the particular use case presented here. LQG-FFB and the online system identification scheme are a first attempt at a suitable combined application.

4 TESTING, RESULTS AND DISCUSSION

This chapter covers testing of the system previously described: methodology for the simulations and experiments carried out, how the collected results compare to initial expectations and which are the main takeaways from the observed outcome.

4.1 Simulations and Experiments

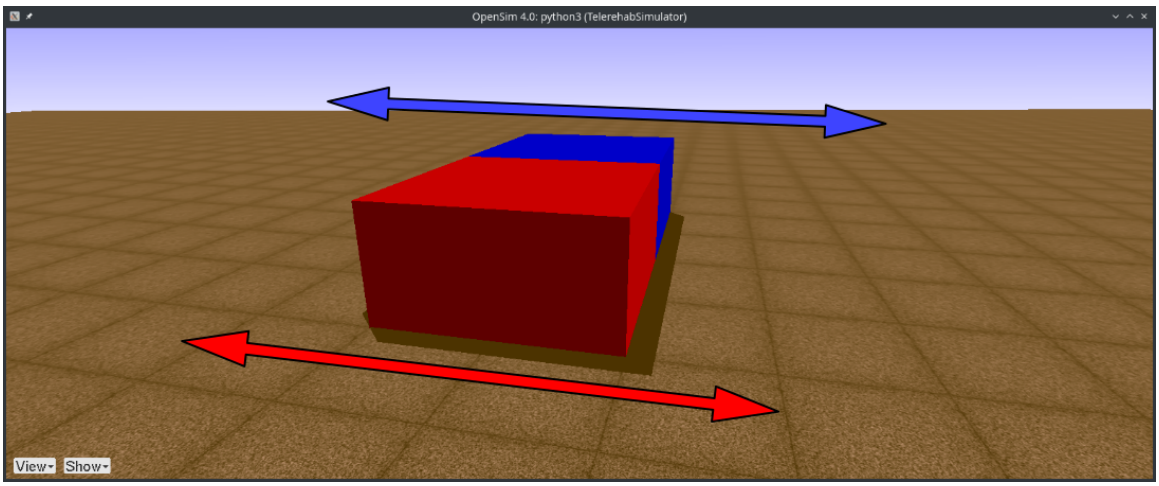
This section describes the gradual approach to test and validate the proposed telerehabilitation subsystems.

4.1.1 *OpenSim simulation*

In order to isolate testing of the algorithms themselves from modeling or delay estimation errors, a virtual simulation was performed. *OpenSim* physics libraries were used to calculate the reaction of two sliding rigid bodies – as user and environment effectors (Fig. 11) – of same inertia ($M_h = M_e = 1kg$) synchronized via bilateral control.

The simulator[1] and controllers were implemented in *Python*, as *OpenSim* provides bindings for that language which allows for practical data plotting with *Matplotlib*[2]. Using *OpenSim* also enables online visualization of the simulated model (Fig. 41).

Figure 41 – Rigid bodies movement shown in the *OpenSim* simulation visualizer.



Source: The author

As a real user is not able to reproduce the exact same motion along different simulations, hindering direct comparison of results, initially a virtual operator was modeled

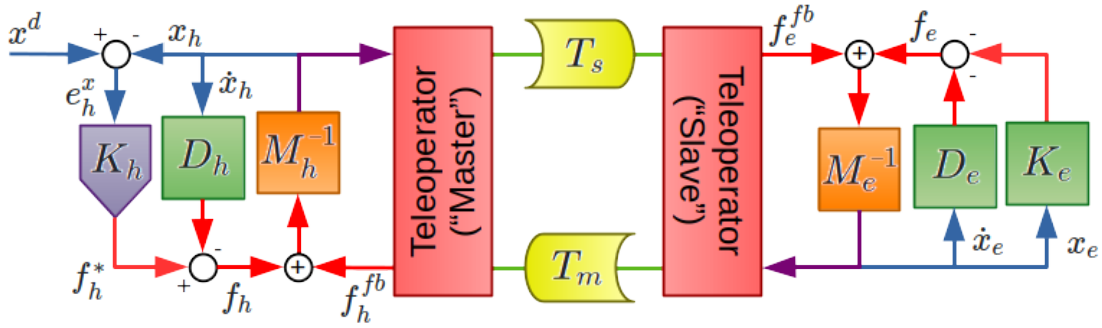
[1] https://github.com/EESC-MKGroup/OSim-Telerehab-Simulator
[2] https://matplotlib.org/

according to Eq. 2.10, to follow a given position reference x^d, randomly generated with maximum frequency of $0.5Hz$. The contributions of robot dynamics, AAN control, and game scaling were not taken into account, so that f_h is equivalent to human-side f^{ext}.

For evaluation of both teleoperation algorithms – wave ($\lambda = 0.5$) and LQG ($\rho = 0.0001$) based – in different conditions of remote user/environment reaction force f_e and impedance (K_e, D_e), 4 configurations of motor interaction based on definitions from Atashzar, Shahbazi, et al. (2017) were simulated:

- Passive-resistive (Fig. 42): master-slave interaction with environment of spring-damper behaviour ($K_e = 8N/m$ and $D_e = 2N \cdot s/m$), where the virtual user at master side has $K_h = 4N/m$ and $D_h = 4N \cdot s/m$, representing a patient with low muscular force (proportional gain) and high spasticity (damping).

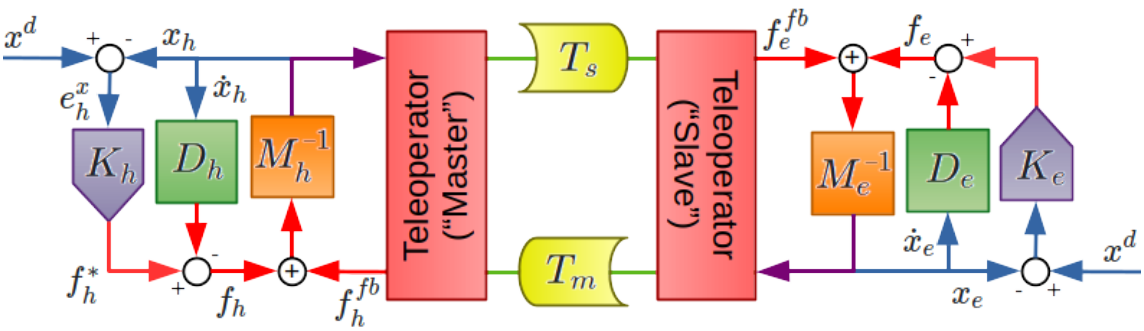
Figure 42 – Diagram of teleoperation simulation with passive-resistive remote environment.



Source: The author

- Coordination-assistive (Fig. 43): two virtual users cooperate to follow the same reference, keeping the same master virtual user and making slave side work as a second one ($K_e = 8N/m$ and $D_e = 2N \cdot s/m$) with the same x^d, representing a helping healthy therapist. This represents the expected telerehabilitation use case that can violate the passivity constraint at times.

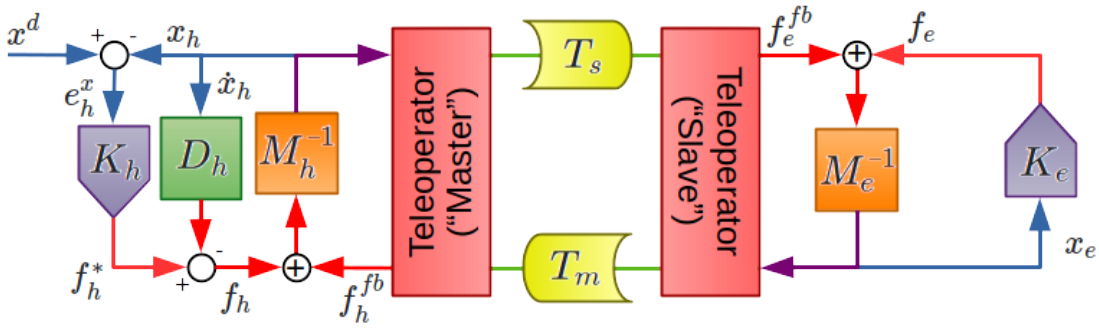
Figure 43 – Diagram of teleoperation with coordination-assistive remote user.



Source: The author

- Power-assistive (Fig. 44): slave amplifies master motions, keeping the same master virtual user and setting the slave to react with a negative impedance of $D_h = -2N \cdot s/m$. This represents the worst-case scenario when the remote environment is constantly adding energy to the system.

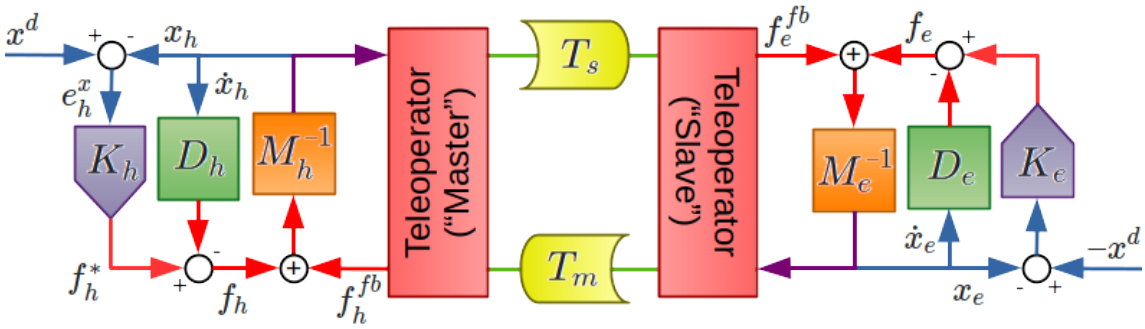
Figure 44 – Diagram of teleoperation with power-assistive remote user.



Source: The author

- Active-resistive (Fig. 45): same setup as the coordination assistive one, but inverting the reference for the second (slave) user ($x_e^d = -x_h^d$).

Figure 45 – Diagram of teleoperation with active-resitive remote user.

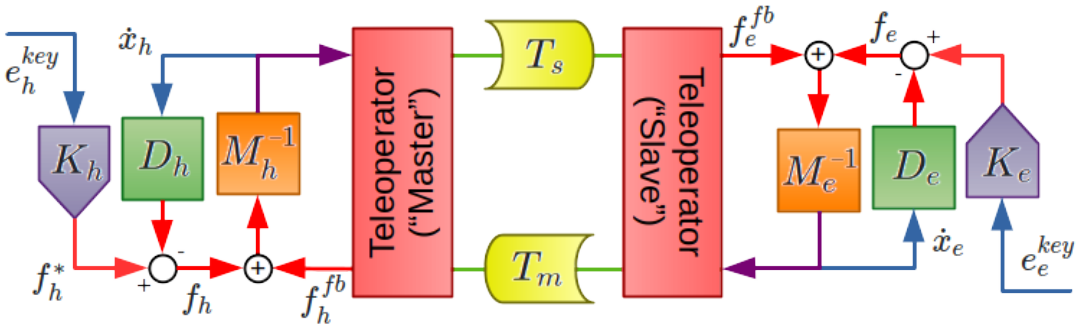


Source: The author

Each simulation run lasted 40 seconds (2000 samples with 0.02s time steps) and was performed with and without signal prediction for every setting to test its effectiveness in compensating delay, artificially applied with varying intensity of 200 ± 100 milliseconds (close to the limit for real-time applications stated in (SUZNJEVIC; SALDANA, 2015)). As terminal impedances were constant and known beforehand, the effects of the passivity controller were tested but online estimation (Eq. 3.3) and adaptation were not evaluated.

To also perform a interactive test, addressing what more closely resembles operator behaviour, the predefined user trajectory was then replaced by real-time user-provided input on master and slave (Fig. 46).

Figure 46 – Diagram of teleoperation with real-time users input.



Source: The author

That interaction was limited to keyboard key presses, detected using *OpenSim*'s visualizer window functions, interpreted as position error (e_h^{key}, e_e^{key}) and used to generate active force input signals f_h^* and f_e^* with a slightly smoother form:

$$f^*_{h\,k} = 0.8 f^*_{h\,k-1} + 0.2 K_h {e_h^{key}}_k, \quad f^*_{e\,k} = 0.8 f^*_{e\,k-1} + 0.2 K_e {e_e^{key}}_k, \quad e^{key}_{h,e} = \pm 1. \tag{4.1}$$

Both teleoperators were run under the same variable delay of previous simulations, with $K_h = 4N/m$ and $K_e = 8N/m$. As the effects of active assistance or resistance are already covered, this test involved no external reference trajectory x^d.

4.1.2 *Laboratory Experiments*

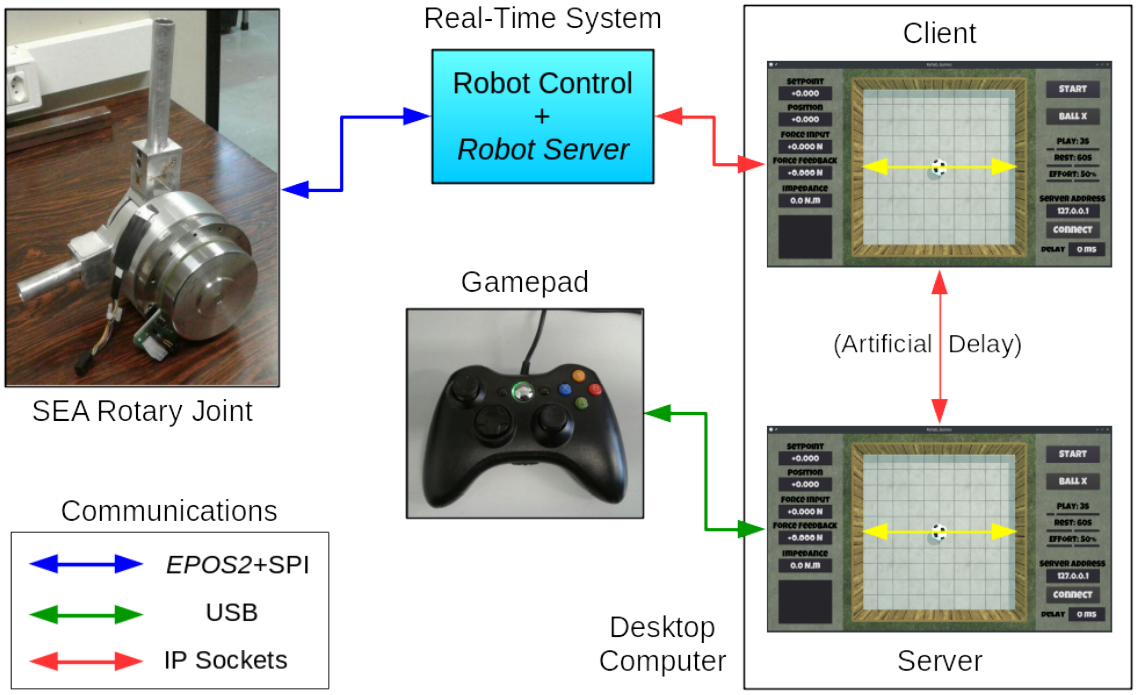
Simulation alone would not be enough to demonstrate success for any tried controller, as virtual models will not account for many unknown nonlinearities and sources of disturbance and noise. On the other hand, directly going for experiments with actual rehabilitation hardware and human subjects is risky and time consuming, as expertise on therapy procedures and networked force reflection systems would be required.

4.1.2.1 Testing platform

Thus, a minimal and safe test bench was built first at *ReRob* for real tests, simulating the interaction between an orthosis-wearing patient – playing at a game client ("master") – and a joystick/gamepad-equipped doctor – playing at server ("slave"). A rotary joint was attached to one of the described SEAs, with its links working as handles (Fig. 20a), in order to allow manipulation by a single person.

In that simplified testing layout, described in Fig. 47, all game instances run on a single desktop PC, with enough processing power to not hinder the timing of physics and network updates. Green and blue colors are used to indicate specific controller (active joint and gamepad) connections, while yellow arrows show the virtual coordinates affected by force exchanges.

Figure 47 – Layout for simplified test of bilateral telerehabilitation game.



Source: The author

Communications in red between game client and server are where delay effects are mostly concentrated. In a multi-client setup, information exchange between them would be affected by two delayed transmissions, but that is less significant as the semi-authoritative server is the node responsible for state synchronization across the system.

4.1.2.2 Testing routine

For the experiment, during 50 ($N = 2500$ samples) seconds of lagged teleoperation, healthy subjects handled the SEA and the gamepad following similar game targets (for the same axis on *Move Ball*). Robot user had to switch between tighter and more relaxed grips, to observe the effects of damping adaptation in isolation. Master local control was set to full compliance ($f^d = 0$), so that f^{int} is expected to correlate to user input f_h.

Inside the gaming machine, artificial network delay – 100 ± 50 ms (25% correlation to previous package) with 5% packet loss – could be emulated, on *Linux*, using the *netem*[3] traffic control (*tc*) utility on the loopback (*lo*) interface:

```
# tc qdisc add dev lo root netem delay 100ms 50ms 25% loss 5%
```

Code 4.1 – Command-line arguments for *tc*'s *netem* packet delay and loss emulation on the loopback (*lo*) network interface.

[3] https://wiki.linuxfoundation.org/networking/netem

Test rounds were held for teleoperation algorithms showing the best performance in simulations, with and without impedance adaptation. Preliminary offline calibration also set the lower bound for patient's impedance, taken as the average of 1000 results from least squares estimation (Eq. 3.3) with no intentional active user input f_h^* and sinusoidal robot force setpoint f_r^d.

4.1.3 *Data Logging and Evaluation*

To evaluate feasibility and performance of any proposed delay compensation or AAN approach, being able to register relevant experimental data, for later comparison, is essential. The following were the synchronization variables logged during operation on both terminals of the telerehabilitation setup:

- Actual and reference/setpoint positions;
- Input, human-robot interaction and feedback force values;
- Network delay, estimated as half of low-pass filtered RTT on laboratory experiment.

On all tests, those variables were registered and indexed by a common time reference: simulation steps for the first case and system clock for the second.

As the controlled virtual objects in both simulator and game are pure inertias, EOP is assessable as difference between input work and total kinetic energy (net work), via discrete integration over the N time steps. That way, a unified *passivity measurement* (PM), for which negative values mean terminal nonpassivity, is established for comparison with a reference case of direct contact ($T = 0$):

$$\sum_{k=0}^{N}(f_k^{ext} + f_k^{fb})\dot{x}_k\Delta t = \frac{M_h\dot{x}_k^2}{2} \implies \text{passivity} : PM = \sum_{k=0}^{N} f_k^{ext}\dot{x}_k\Delta t - \frac{M_h\dot{x}_k^2}{2} \geq 0. \quad (4.2)$$

Transparency between teleoperation sides was verified by calculation of *root mean square* (RMS) differences between locally and remotely perceived slave-side impedance Z_e (Fig. 11). It is also proposed here to combine RMS errors for velocity tracking and force reflection into a single *transparency measurement* (TM) to be minimized:

$$\text{transparency} : TM \to 0 \implies \sqrt{\frac{1}{N}\sum_{k=0}^{N}(\dot{x}_h - \dot{x}_e)_k^2} + \sqrt{\frac{1}{N}\sum_{k=0}^{N}(f_h^{ext} - f_e^{fb})_k^2} \to 0. \quad (4.3)$$

In the real test, to find out if the model of human input (Eq. 2.10) works for online calculation of player's biomechanical properties (K_h, D_h, M_h), averages of $\hat{K}_h$ and $\hat{D}_h$ during operation phase ($\hat{K}_{op}^{avg}$, $\hat{D}_{op}^{avg}$) are compared to the offline calibration ones ($\hat{K}_{cal}^{avg}$,

$\hat{D}_{cal}^{avg}$). Correlation between $\hat{K}_h$ and f^{int} is also observed to evaluate if the first is a proper indicator of user effort:

$$corr(f, \hat{K}) = \frac{1}{N}\sum_{k=0}^{N} |f^{norm}\hat{K}_h|_k, \qquad f^{norm} = \frac{f^{int}}{f^{max}}, \qquad f^{max} = \sup_{t\in[0,T_{op}]} |f^{int}(t)|. \tag{4.4}$$

In order to avoid huge oscillations, some limits are defined for measured interaction forces and estimated impedances (expressed in inertia, damping and stiffness), displayed in Tab. 3. Inertia is assumed to remain mostly constant, while user active stiffness is calculated even during calibration phase, despite being expected to remain near 0.

Table 3 – Force and impedance hard limits during calibration and operation phases.

	f^{int}	$\hat{M}_h$	$\hat{D}_h$	$\hat{K}_h$
Minimum (calibration)	-	$0.1\ kg$	$0.0\ N \cdot s/m$	$0.0\ N/m$
Minimum (operation)	-	$\hat{M}_{cal}^{avg}$	$\hat{D}_{cal}^{avg}$	$\hat{K}_{cal}^{avg}$
Maximum (all phases)	$10.0\ N$	$2\hat{M}_{cal}^{avg}$	$100.0\ N \cdot s/m$	$100.0\ N/m$

Source: The author

4.2 Discussion of Results

In this section, the registered outcome from performed tests is presented and analyzed according to aforementioned criteria.

4.2.1 *Predefined Reference Trajectory and Simulated Users*

The usage of a virtual user in *OpenSim* simulator, although not ideal for representing a real world scenario, enabled direct comparison of various teleoperator configurations due to repeatability. Thus, the most comprehensive algorithm comparison was done using this setup, displaying performance data in compact table format. Most relevant results are highlighted in bold characters.

4.2.1.1 Passive-resistive environment

Results for remote interaction with a spring-damper environment in Tab. 4 reveal the difficulty to reproduce stiffness K_e perception, as studied in (HIRCHE; BUSS, 2012), exhibited by its higher RMS deviation.

In any case, the LQG-based teleoperator achieved better overall transparency (indicated by TM) when no prediction or passivity control was involved. Analyzing each impedance component separately, however, shows that the better K_e mirroring comes at the cost of worse inertia M_e and damping D_e tracking, besides eventual passivity violation. Wave variables solutions, in turn, lead to a PM slightly closer to the reference case.

Table 4 – Results for simulated interaction of virtual master user with passive-resistive environment (Fig. 42). Reference best-case PM: $\mathbf{0.032}N \cdot m$.

	Tracking error	Inertia error	Damping error	Stiffness error	PM	TM
Wave variables (no prediction)	0.006	0.613	1.726	4.476	0.046	0.106
LQG-FFB w/o M-TDPC (no prediction)	0.010	0.269	1.543	3.311	0.011	**0.070**
LQG-FFB w/ M-TDPC (no prediction)	0.025	0.150	0.404	6.121	0.046	0.094
Wave variables (prediction)	0.007	0.865	0.930	3.993	**0.032**	0.119
LQG-FFB w/o M-TDPC (prediction)	0.010	0.214	1.693	3.275	-0.004	0.082
LQG-FFB w/ M-TDPC (prediction)	0.025	0.142	0.495	6.174	0.045	0.095

Source: The author

4.2.1.2 Coordination-assistive task

Tab. 5 shows the nonpassive response of cooperation with a remote controller with higher proportional gain (low perceived resistance), which the wave operator is not intended for, apparently making it unable to ensure local passivity without hindering transparency – although not all excess energy was dissipated. In the case of the variants of LQG with direct force transmission, which have overall better performance, it seems like passivity control causes less distortion in impedance perception than signal prediction.

Table 5 – Results for simulated interaction on coordination-assistive task between virtual master and slave (Fig. 43). Reference best-case PM: $\mathbf{-0.300}N \cdot m$.

	Tracking error	Inertia error	Damping error	Stiffness error	PM	TM
Wave variables (no prediction)	0.009	0.738	0.000	1.721	-0.179	0.299
LQG-FFB w/o M-TDPC (no prediction)	0.013	0.268	0.000	0.082	-0.318	**0.121**
LQG-FFB w/ M-TDPC (no prediction)	0.013	0.309	0.000	0.094	**-0.303**	**0.121**
Wave variables (prediction)	0.009	1.083	0.000	1.004	-0.323	0.340
LQG-FFB w/o M-TDPC (prediction)	0.012	0.180	0.066	0.150	-0.495	0.179
LQG-FFB w/ M-TDPC (prediction)	0.014	0.246	0.041	0.170	-0.422	0.162

Source: The author

4.2.1.3 Power-assistive task

In the worst-case scenario of pure amplification (Tab. 6), the need for passivity guarantees becomes evident. Wave-based teleoperators had the best overall performance, while the LQG-based controller could not prevent instability when operating without the passivity regulator. As there was no remote resistance, inertia reflection presented the only significant transparency degradation for the stable systems, where prediction presented some noticeable contribution.

Table 6 – Results for simulated interaction on power-assistive task between virtual master and slave (Fig. 44). Reference best-case PM: **-0.343**$N \cdot m$.

	Tracking error	Inertia error	Damping error	Stiffness error	PM	TM
Wave variables (no prediction)	0.008	0.623	0.577	0.107	-0.047	**0.214**
LQG-FFB w/o M-TDPC (no prediction)	708.47	0.947	0.000	0.004	-3.23e8	2.71e3
LQG-FFB w/ M-TDPC (no prediction)	0.111	0.732	0.060	0.000	-3.265	0.429
Wave variables (prediction)	0.016	0.773	0.000	0.000	**-0.439**	0.255
LQG-FFB w/o M-TDPC (prediction)	7.37e4	0.963	0.000	0.105	-4.29e12	2.32e5
LQG-FFB w/ M-TDPC (prediction)	0.129	0.775	0.000	0.202	-3.477	0.519

Source: The author

4.2.1.4 Active-resistive interaction

When the teleoperators have opposing trajectory goals (Tab. 7), the weaker virtual patient's side ends up performing a negative net work. Somehow analogously to the passive-resistive case, the LQG-based solutions had poor stiffness transmission and energy balance, especially when passivity control is being used, but as its absence does not lead to instability there is room for adaptively relaxing the constraint.

In the end, wave teleoperators clearly appear better at handling contact behaviour under more critical conditions, while the alternative approach causes less distortion during freer movements.

4.2.2 *User-Provided Keyboard Input*

The other way the simulator was used relied on the window input features of *OpenSim*'s libraries. With that capacity, the same implementation could be tested in response to unpredicted opertator behaviour.

Table 7 – Results for simulated interaction on active-resistive task between virtual master and slave (Fig. 45). Reference best-case PM: **-0.161**$N \cdot m$.

	Tracking error	Inertia error	Damping error	Stiffness error	PM	TM
Wave variables (no prediction)	0.017	0.529	0.000	2.241	-0.168	0.166
LQG-FFB w/o M-TDPC (no prediction)	0.026	0.199	0.305	8.642	-0.067	**0.085**
LQG-FFB w/ M-TDPC (no prediction)	0.035	0.411	3.650	22.397	0.007	0.131
Wave variables (prediction)	0.019	0.451	0.000	2.646	**-0.162**	0.169
LQG-FFB w/o M-TDPC (prediction)	0.025	0.186	0.247	4.553	-0.094	0.104
LQG-FFB w/ M-TDPC (prediction)	0.036	0.403	2.240	16.019	0.005	0.131

Source: The author

When dealing with actual user interaction, with sharper (although smoothed) force input changes, delivering proper impedance reproduction would depend on the ability to respond to higher frequency inputs.

As less experiment rounds were performed for that setup, and repeatability is not entirely possible, results are presented in a more visual manner and qualitatively evaluated. Values registered over the course of operation were plotted in 4 graphics:

- Position synchronization: comparing trajectories at master (blue line) and slave (red line) with one another and in relation to resulting movement in the ideal condition of zero-delay direct contact (dotted black line);

- Force exchange: to visualize how external input forces on both master (green line) and slave (magenta line) are reproduced as feedback at the opposite side (blue and red lines, respectively);

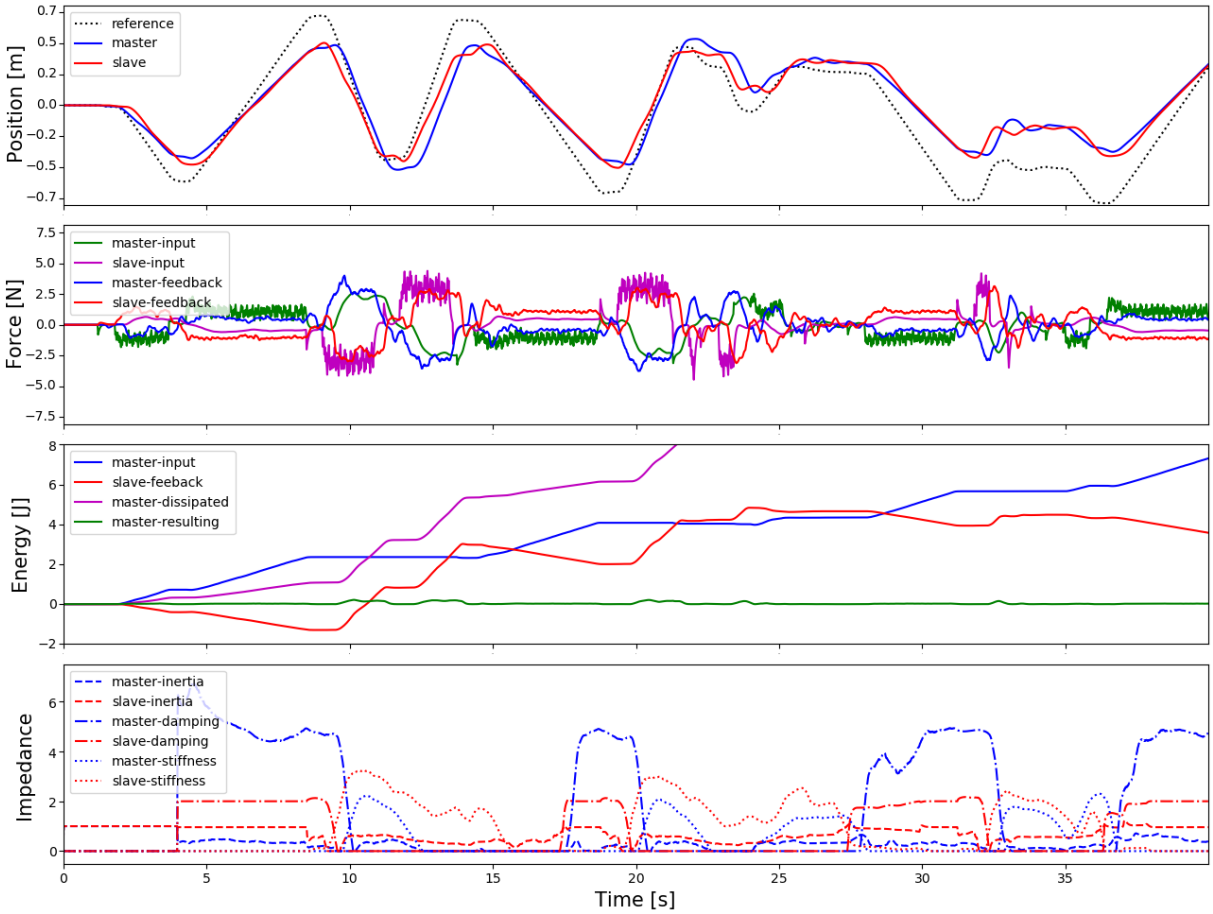
- Energy balance: presenting for master side the integrals of mechanical power delivered by local force input (blue line) and remote feedback (red line), as well as energy dissipated by local damping (magenta line) and resulting net work (green line), whose final value is equivalent to PM;

- Transparency performance: showing the matching between values perceived from master side (blue lines) and verified on slave (red lines) for the impedance/admittance components, namely inertia (dashed), damping (dot-dashed) and stiffness (dotted).

4.2.2.1 Wave variables without prediction

The wave teleoperator (Fig. 48) is able to provide good position synchronization, at the cost of more feedback damping, deviation from reference trajectory and higher impedance distortion. That is expected, as guaranteeing passivity is the main focus of that type of controller.

Figure 48 – Results for interactive simulation of wave teleoperator without signal prediction.

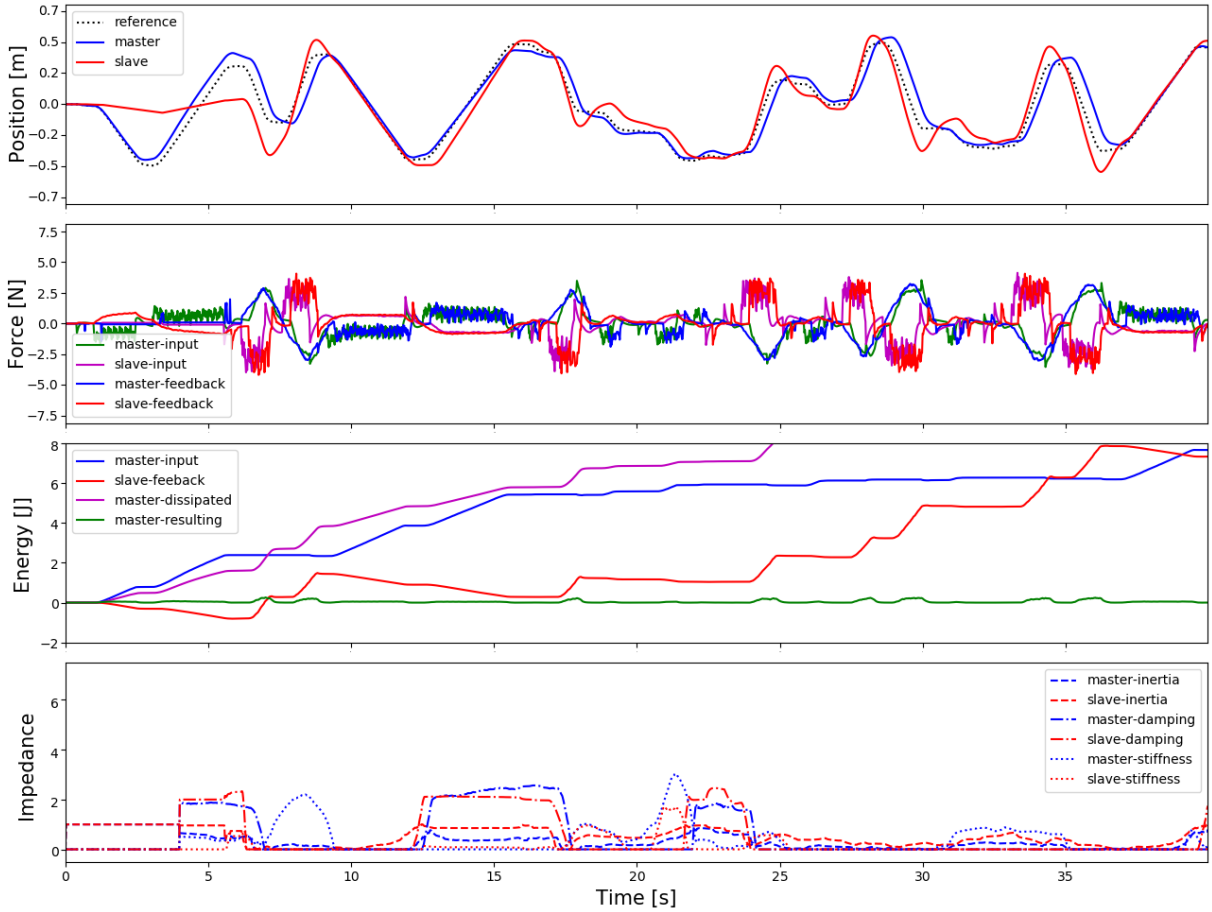


Source: The author

4.2.2.2 LQG-FFB with passivity control and without prediction

The LQG-based solution (Fig. 49), on the other hand, was able to reflect quick oscillations in the input forces more faithfully, and through that presented better transparency and position tracking relative to the reference. One drawback from the usage of the passivity controller is the fact that feedback transmission is poor at the beginning of the operation, due to the way that M-TDPC requires some dissipated energy buffer to work, what could be mitigated by some nonzero initial tolerance.

Figure 49 – Results for interactive simulation of LQG teleoperator without signal prediction.



Source: The author

4.2.2.3 Wave variables with prediction

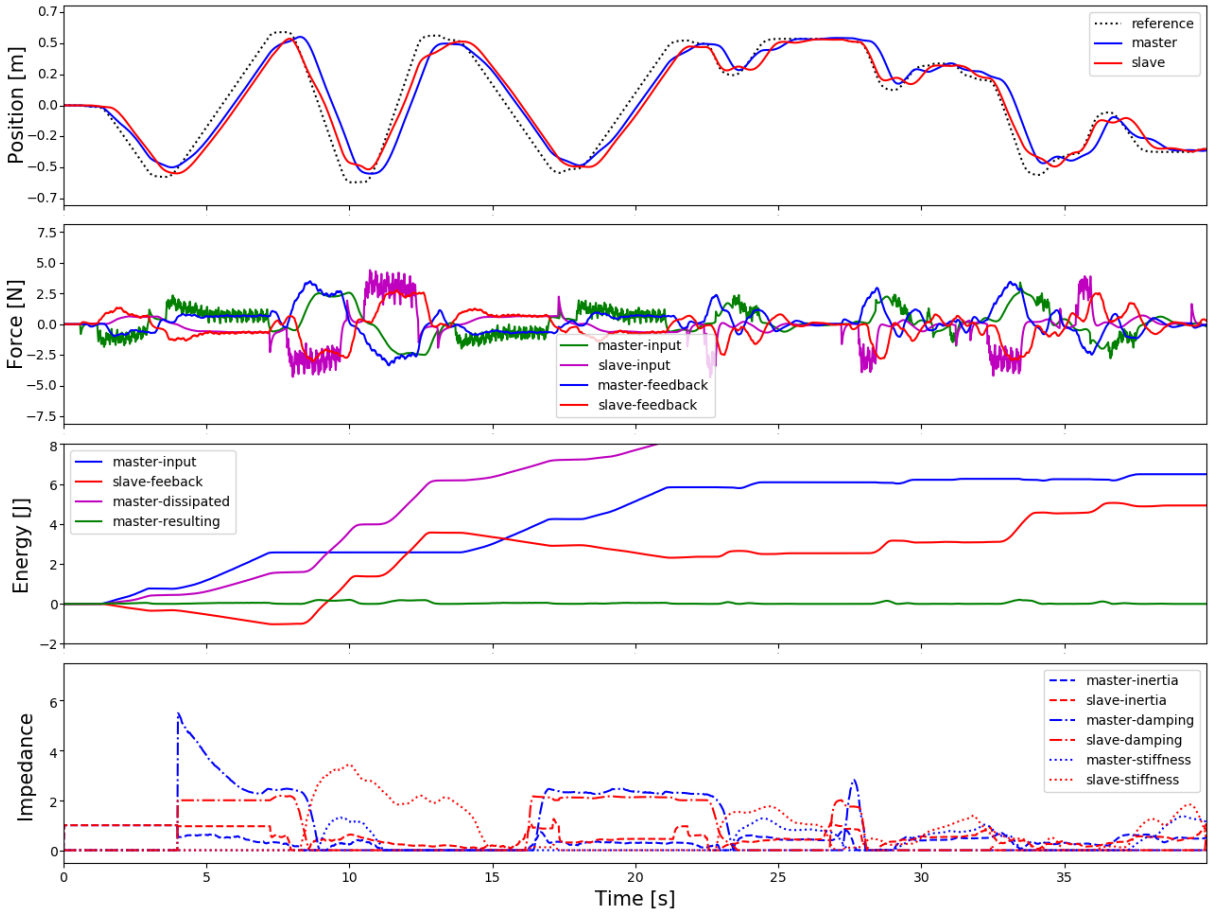
The application of *Kalman* prediction to wave variables (Fig. 50) showed good results, improving position tracking and impedance matching between both transmission sides. The only visible caveat is the poor reproduction of stiff behaviour.

4.2.2.4 LQG-FFB with passivity control and prediction

Combining prediction to LQG-FFB (Fig. 51) did not add much to performance. Slightly better overall transparency was achieved at the cost of little overshoots in position and impedance tracking. That trade-off suggests the extrapolation algorithm has some room for fine-tuning.

It should be noted that resulting work output (inertia's kinematic energy) remained bounded, thus preserving stability of the system, for every tested algorithm.

Figure 50 – Results for interactive simulation of wave teleoperator with signal prediction.



Source: The author

4.2.3 *Game and Robot Controller Experiments*

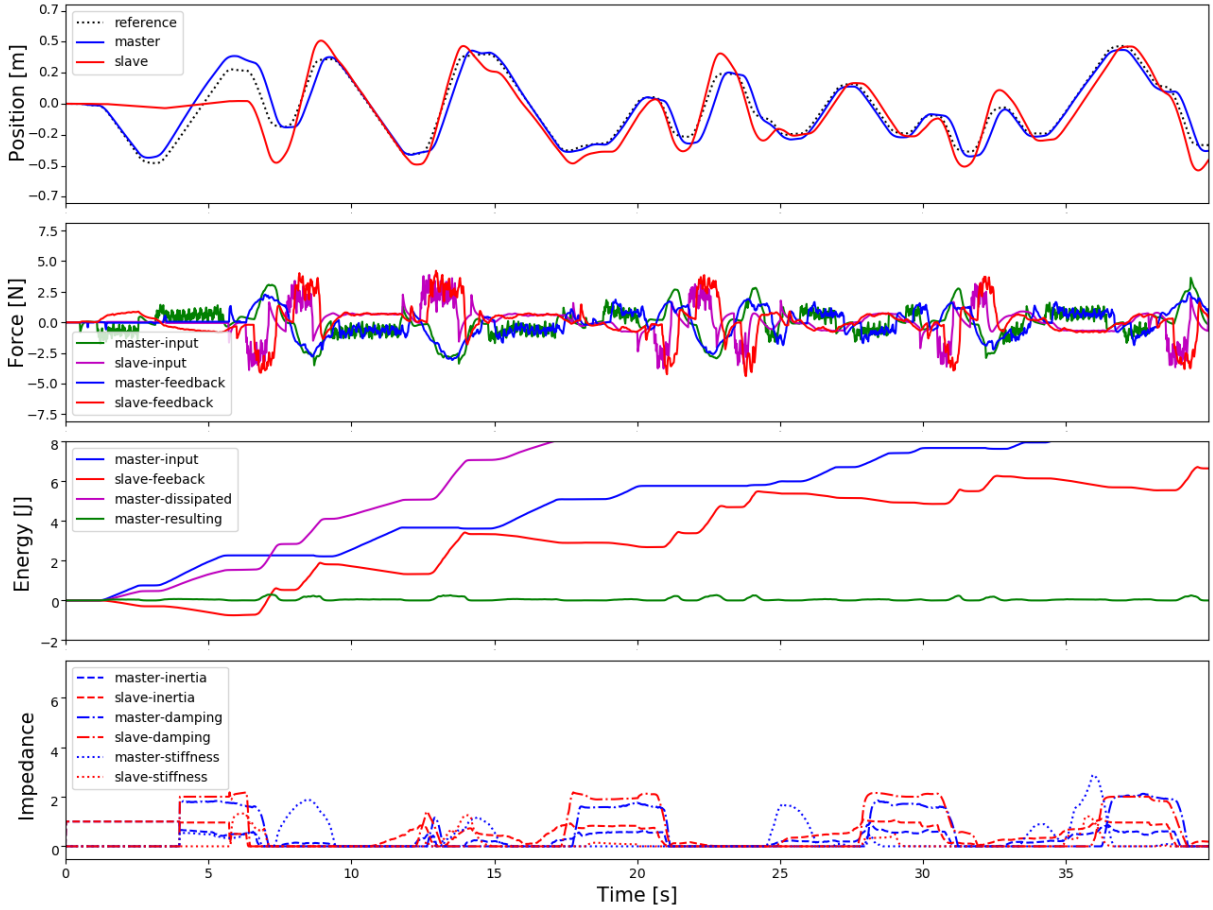
The actual robot teleoperation and gaming test results were separated in 2 aspects: comparison of offline versus online mechanical system identification and performance assessment of the most promising algorithms from simulation Sections 4.2.1, 4.2.2. Data from both are presented in a mix of tabular and graph formats.

4.2.3.1 System calibration and online identification

Tab. 8 lists some of the most significant impedance – damping D_h and active stiffness K_h – estimation results, during calibration and operation phases. The last column also shows correlation factor between $\hat{K}_h$ and measured human-robot interaction force f^{int} for the same testing rounds. Dimensional units are being used according to Tab. 2.

During what was basically a coordination assistance (collaborative) interaction, due to sharing of position targets, the robot-attached player was able to perform the task in a stable manner even when online adaption set model parameter values greater than the ones taken offline.

Figure 51 – Results for interactive simulation of LQG teleoperator with signal prediction.



Source: The author

Table 8 – Averages of estimated impedance values and stiffness-force correlation.

	$\hat{K}_{cal}^{avg}$	$\hat{K}_{op}^{avg}$	$\hat{K}_{op}^{avg}$ / $\hat{K}_{cal}^{avg}$	$\hat{D}_{cal}^{avg}$	$\hat{D}_{op}^{avg}$	$\hat{D}_{op}^{avg}$ / $\hat{D}_{cal}^{avg}$	$corr(f, \hat{K})$
Test round 1	5.491	9.317	1.697x	4.228	5.238	1.239x	1.246
Test round 2	3.024	5.534	1.830x	1.469	2.992	2.037x	1.360
Test round 3	2.994	6.957	2.323x	1.462	2.338	1.600x	1.458
Test round 4	2.376	6.641	2.795x	2.090	2.446	1.170x	1.124

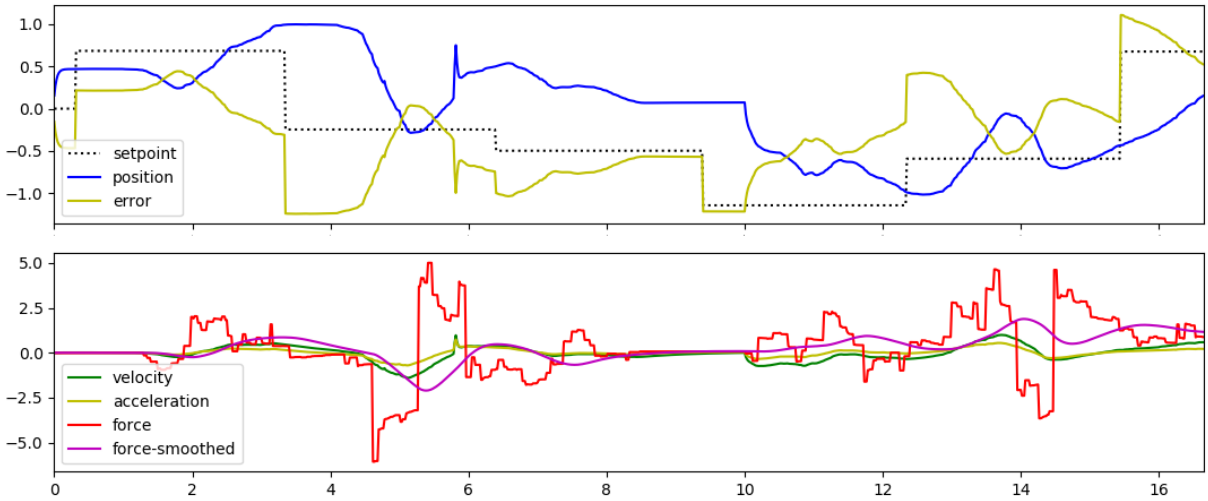
Source: The author

Nonetheless, the linearization method did not identify much greater impedance values when the user was actively following targets. Based on Atashzar, Shahbazi, et al. (2017), it was expected that committed operators show up to 500% increase in muscular damping.

Also, higher increases in perceived stiffness do not appear to lead to better correlation with interaction forces. However, experience during gameplay made noticeable that f^{int} readings were oscillating significantly, partly due to joint fitting issues.

That impression was reaffirmed by analysis of relationship between player position tracking and force inputs, presented in Fig. 52. When force signals are smoothed out, their correlation to user acceleration is noticeable, but that would not be a viable solution during operation because of the introduced phase shift.

Figure 52 – Data sample of player/effector position tracking, resulting velocity/acceleration and measured interaction forces during gameplay.



Source: The author

4.2.3.2 Teleoperators performance

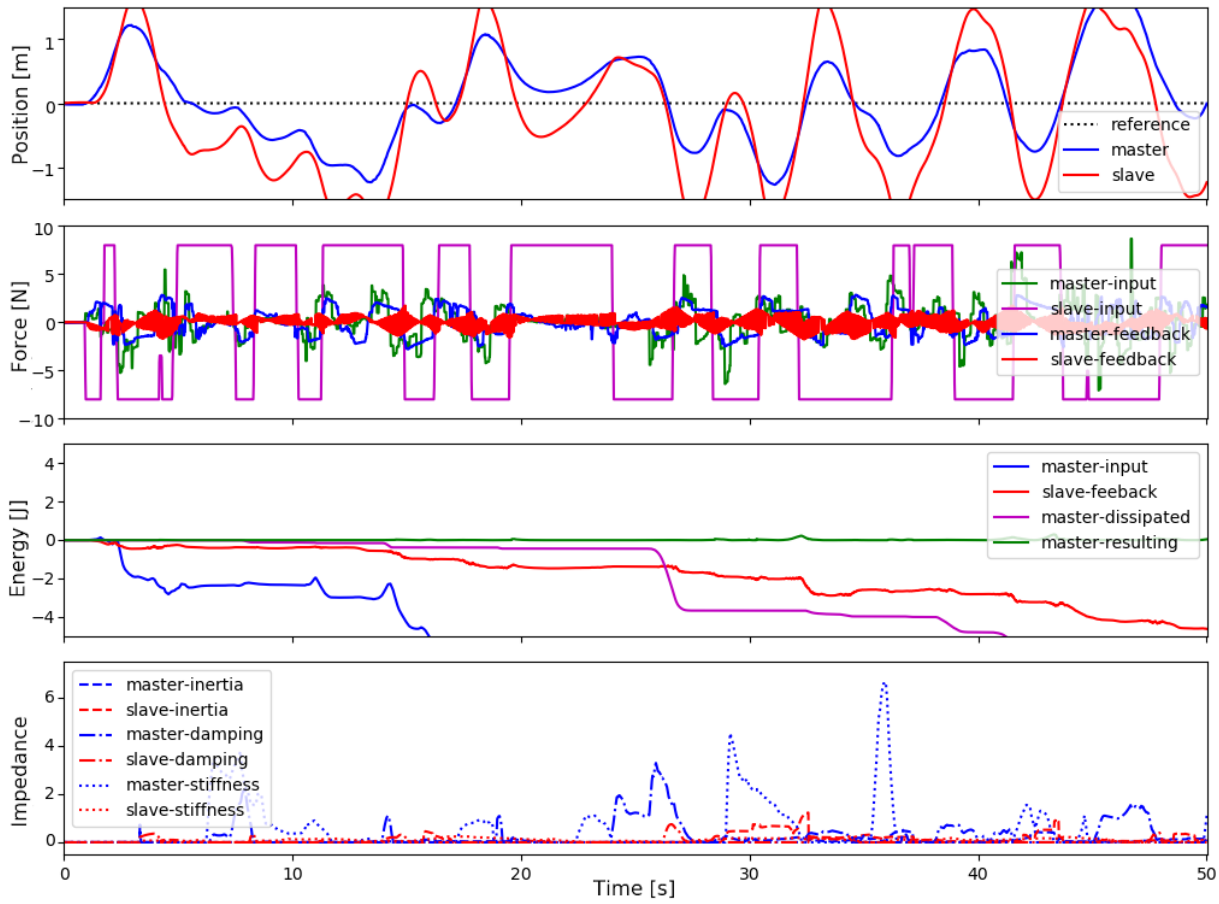
The same performance measurements from Section 4.2.1 were also taken for the game experiments described in Section 4.1.2, in order to compare them more objectively, and are listed in Tab. 9. Most relevant results are highlighted in bold characters.

4.2.3.3 Wave variables with prediction

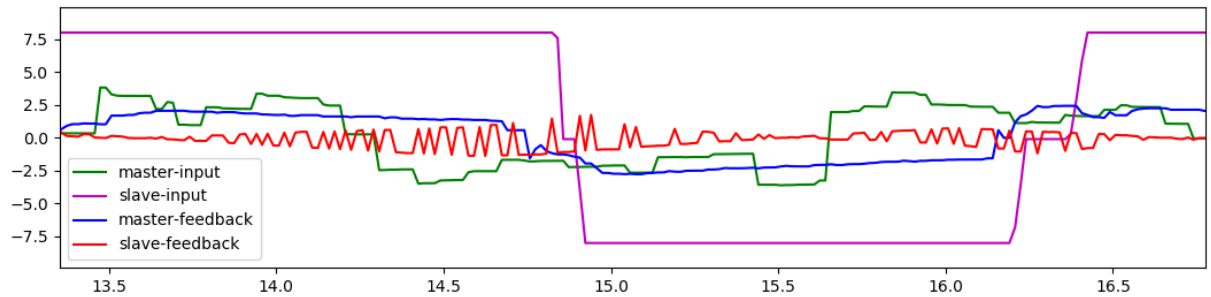
The application of wave variables with signal prediction presented moatly expected results, displayed here in similar fashion to Section 4.2.2. For the non-adaptive case in Fig. 53, low b values calculated just from offline calibration allowed to much trajectory deviation and even violations of the energy balance. Slave force feedback seems to follow a beat frequency pattern, most likely due to incomplete filtering of the wave reflection effect.

Figure 53 – Results for multiplayer game with wave teleoperator with signal prediction and impedance adjusted from offline calibration: (a) complete experiment data and (b) detail of force exchange signals.

(a)



(b)



Source: The author

Table 9 – Performance metrics for teleoperation algorithms tested in multiplayer game.

	Tracking error	Inertia error	Damping error	Stiffness error	PM	TM
Wave variables (offline calibration)	0.526	0.169	**0.557**	1.049	-9.156	2.436
Wave variables (online adaptation)	0.351	0.118	1.545	0.995	9.873	2.545
LQG-FFB w/ M-TDPC (offline calibration)	**0.258**	**0.029**	0.891	**0.208**	4.824	1.505
LQG-FFB w/ M-TDPC (online adaptation)	0.313	0.031	1.032	0.963	2.391	**1.473**

Source: The author

When updating the wave impedance with higher damping values estimated online (Fig. 54), position tracking and passivity were improved in comparison. That came at the cost of bigger amounts of damping and force distortion, though, to the point that there was no improvement in the already hindered transparency.

Another drawback was the fact that the wave prediction teleoperator displayed some steady-state position drift between terminals, but that went practically undetected during active gaming.

4.2.3.4 LQG-FFB with passivity control

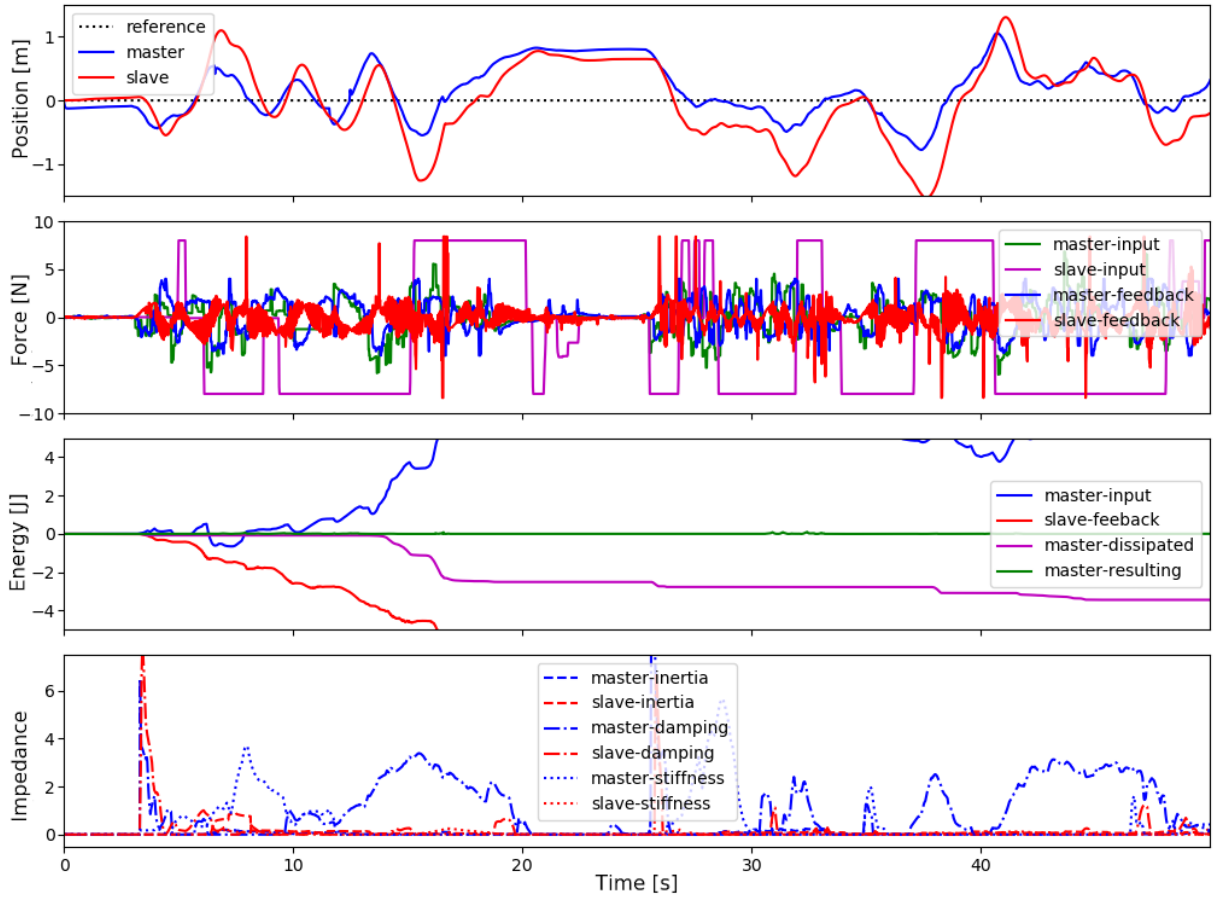
More unexpected were the results for the proposed LQG-based control with force feedback M-TDPC stabilizer (Figs. 55,56) – not using prediction, as it degraded synchronization quality during simulations. Even if the usage of dynamic impedance calculation (Fig. 56) seemed to enhance force reflection, indicated by smaller PM (Tab. 9), position and impedance tracking were not necessarily improved by it.

Counterintuitively, even if RMS errors for each individual impedance component were increased by this approach, the proposed TM index became slightly smaller, indicating that changes in dynamics were somehow compensated or that the employed measurements (Tab. 9) are not a fully reliable metric.

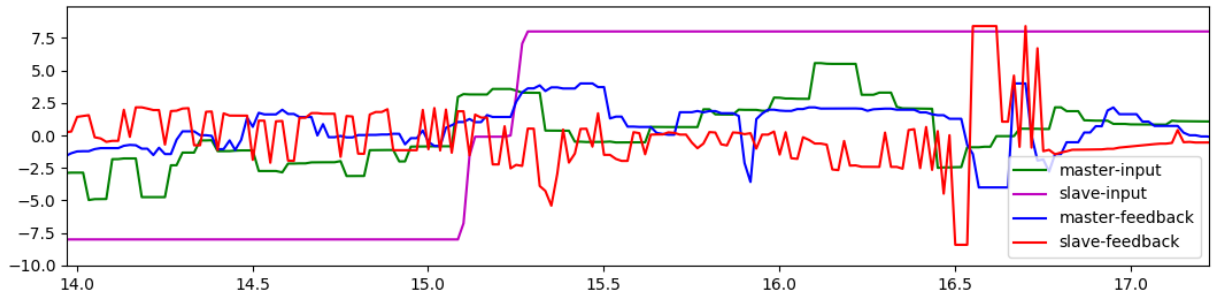
Overall, this technique demonstrated better performance than the wave-based one, regarding position tracking, transparency and maintenance of low excess of passivity still inside the control stability margin. Besides, transmitted forces were smoother (Figs. 55b,56b), which can lead to a noticeably more comfortable user experience.

Figure 54 – Results for multiplayer game with wave teleoperator with signal prediction and impedance adjusted from online estimation: (a) complete experiment data and (b) detail of force exchange signals.

(a)



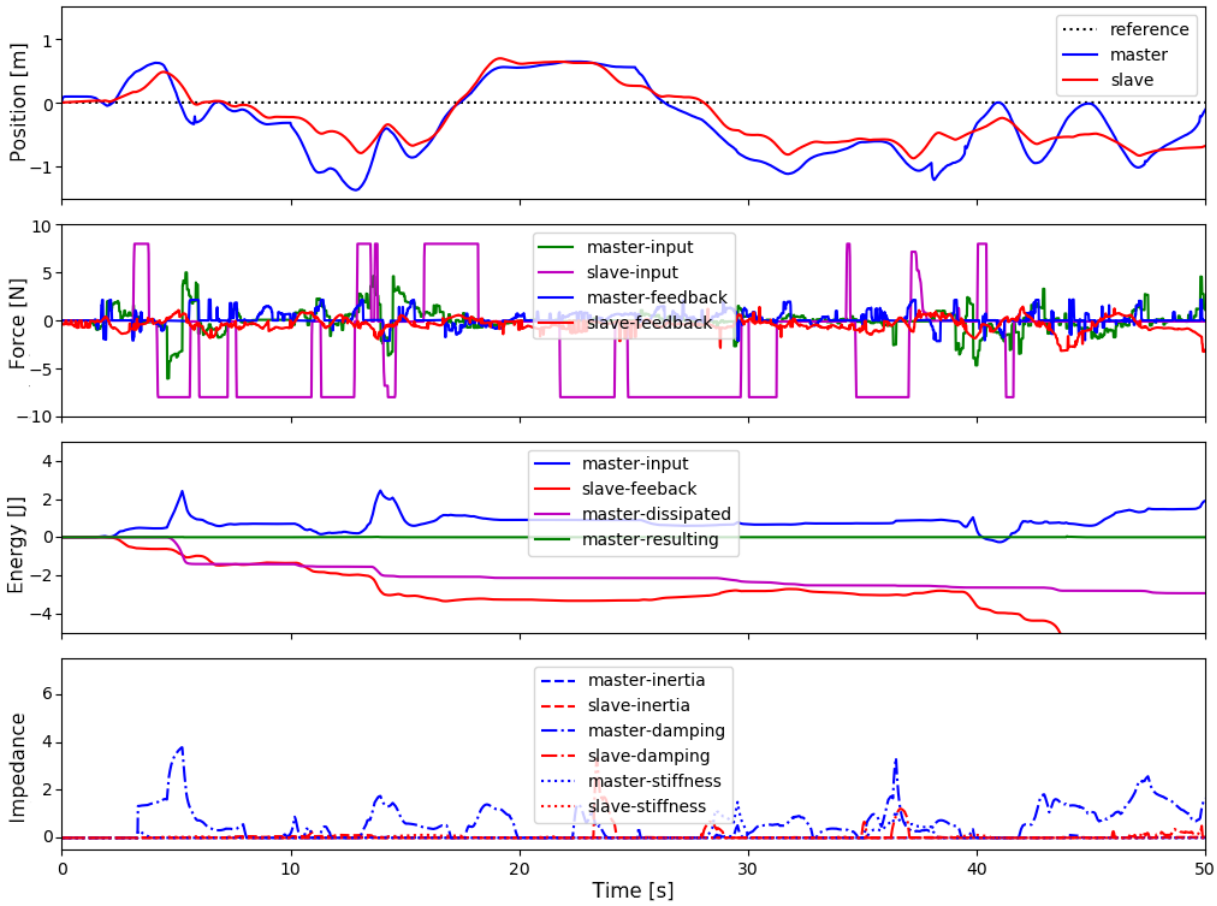
(b)



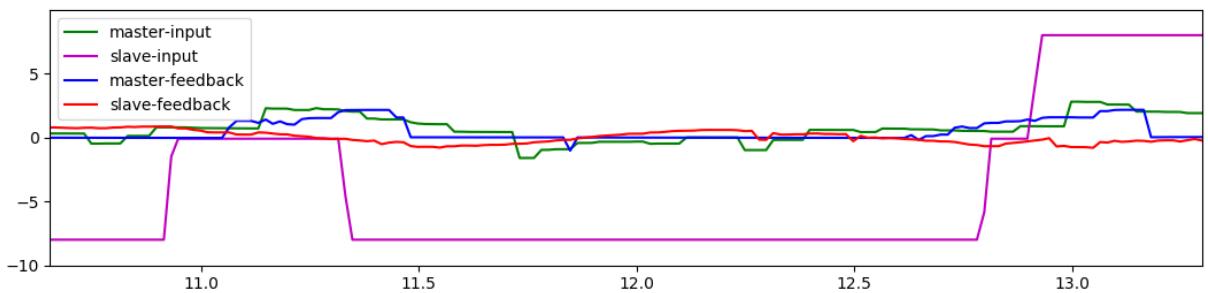
Source: The author

Figure 55 – Results for multiplayer game with LQG-FFB control with M-TDPC and impedance adjusted from offline calibration, under: (a) complete experiment data and (b) detail of force exchange signals.

(a)



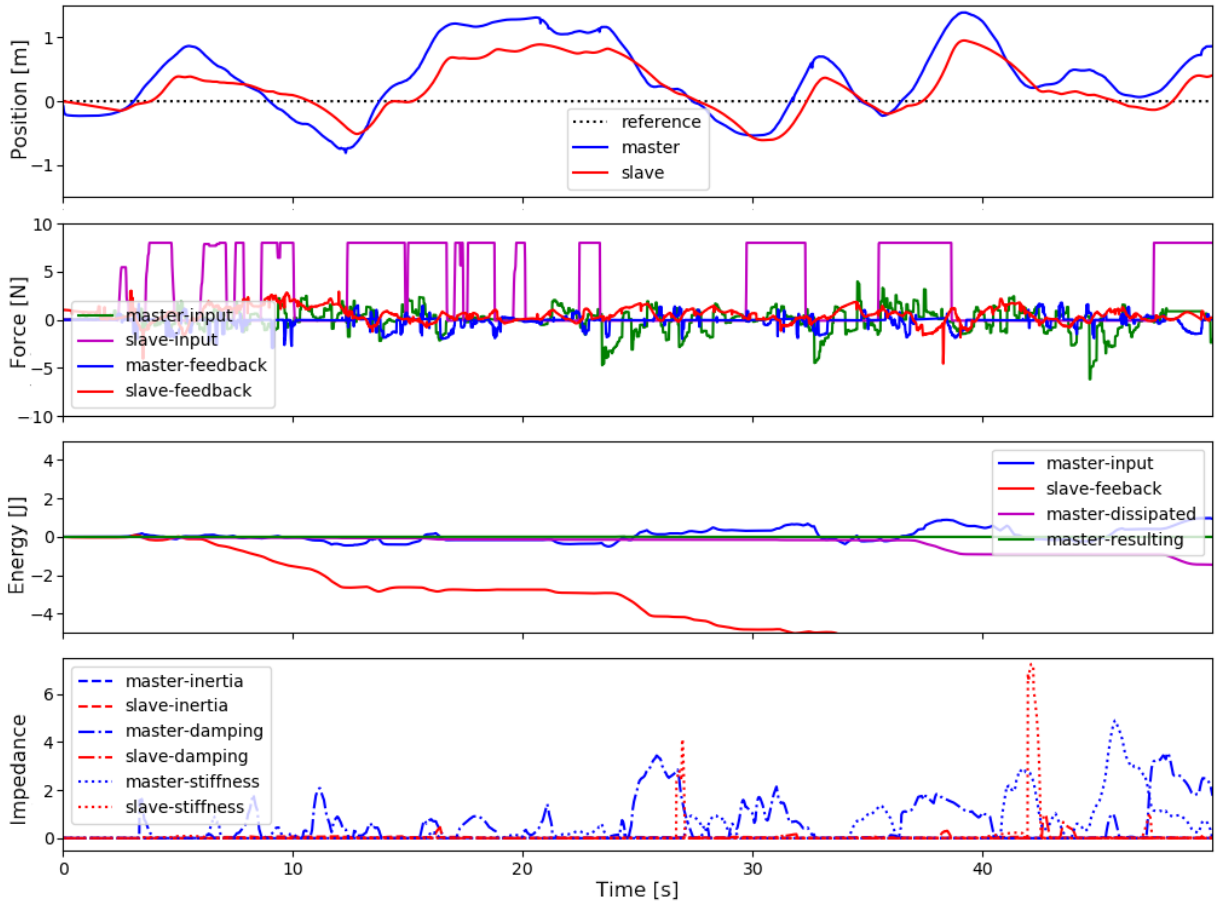
(b)



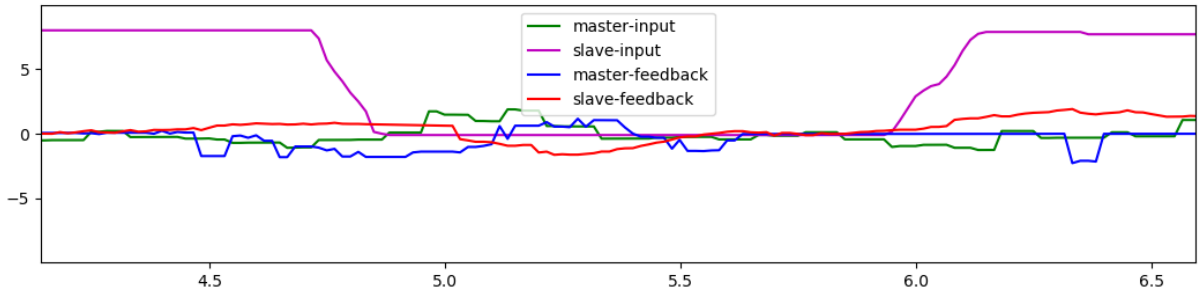
Source: The author

Figure 56 – Results for multiplayer game with LQG-FFB control with M-TDPC and impedance adjusted from online estimation, under: (a) complete experiment data and (b) detail of force exchange signals.

(a)



(b)



Source: The author

4.3 Closing Remarks

The evaluation procedures and results presented here were intended for studying the effects of different variables and operation scenarios in isolation. However, actual remote therapy use cases would be a more nuanced mixture of those conditions, most probably involving 3 or more game participants (network peers). Thus, the system capacity to handle its designated task while subject to less predictable external interferences is still not extensively understood.

5 CONCLUSION

In this thesis, we have dealt with the problem of bilateral and, by extension, multilateral robotic teleoperation for home or group rehabilitation, when applied for simple remote interaction or inside cooperative, collaborative or competitive games, particularly under the circunstance of significant communication delays.

The main goal of any modality of neuromotor therapy is teaching back the compromised body movement, achieved through a combination of high intensity patient training and external corrective stimulus. That outside guiding – assistance or resistance – information is made effective when delivered to the impaired exerciser at the right time and in a comprehensible manner (e.g. without cognitive overload).

From the control theory perspective, it is generally understood that overall system – terminals and communication channel – passivity is a requirement for stable behaviour, but there are disagreements in literature over usage of "weaker" (more restrictive) and "stronger" (less restrictive) formulations of that feasibility criterion.

For assistive teletherapy, trying to ensure a negative energy balance – positive operator's net work, dissipative robotic forces – in each isolated local subsystem is counterproductive, since it nullifies the desired power amplification (by the therapist or task partner) on patient's side. However, analyzing global passivity in real-time is always limited by network latency on data acquisition and the immeasurable nature of human input, so that more permissive approaches end up having to resort to setting stability margins based on dynamic interaction models to improve transparency.

On the other hand, the successful increase in participants' motivation by multiplayer rehabilitation games is conditioned to proper difficulty balancing between them. Assessing player's performance, in order to provide a proportional challenge, also depends on indirect estimations from reference tracking error, activity score, biometric data, etc.

As one of the desirable features of home-based rehabilitation solutions is the usage of low-cost equipment (DARZI; GORŠIČ; NOVAK, 2017), being able to take all information about user commitment from the active device itself – with no dependency on extra hardware – is a sensible design decision. Because muscular effort correlates to changes in limb's mechanical impedance, local adaption algorithms may use the same identification results applied for bilateral force reflection enhancement.

That unified assistance-level, teleoperation-level and game-level control strategy was thereby attempted in this work via a mixture of iterative linear-quadratic regulators, *Kalman* predictors and energy filters, in the hopes of achieving optimal stable transparency. That approach was confronted with a more conservative wave scattering technique to

evaluate the extent of potential improvements over established solutions.

For preliminary simulations, it was seen why the usage of wave transformations is so widespread: they excel at typical remote exploration conditions of hard contact, and do not present huge performance degradation in supposedly non-ideal use cases. We understand that energy exchanges with active environments could work as a superposition of two master-slave setups, still ensuring overall stability.

However, when dealing with types of physical interaction more common in telerehabilitation, like coordination assistance where passivity violations are generally not enough to render the system unstable (not least because any human operator has some intrinsic bodily stabilization capacity), a better stability-reflection compromise may be achieved with algorithms with more relaxed passivity constraints, as the ones presented in this work.

Also, it has been found out that some additional approaches tested here like signal prediction using *Kalman* filters, even if not always advantageous, might be used as an incremental improvements over wave scattering, mainly for systems already modeled around this solution.

The subsequent tests using actual human-machine interfaces and custom games confirm the previous findings, with the caveat that even the transmission of signal integrals will not prevent permanent position drifts when using wave variables for synchronization. Despite the more complex implementation, LQG-FFB also has the advantage of not depending on impedance synchronization between teleoperation terminals.

Surely some experimental results obtained here were suboptimal, partially due to equipment limitations and noisy force measurements. But simulations provided some good indications of outcomes that could be pursued if more robust interaction force estimates could be applied in future works.

Still, some concerns were not yet addressed, leaving room for related research:

- Even though transparent force exchange was studied as a prerequisite for remotely induced learning, the issue of neuromotor re-learning itself remained unresolved. As a statistically significant number of impaired patients (and submission of experiment proposals to an ethical committee) would be necessary for a thorough assessment of generalization ability (WILDENBEEST; ABBINK; SCHORSCH, 2013) after training, that task was left for subsequent developments;

- Similarly, and mostly for the same reasons, no game-level difficulty balancing heuristic was suggested and evaluated. It was assumed for now that local AAN control would suffice for filling the skill gap between distinctly impaired players, but it is possible that adaptions to gameplay mechanics may improve upon it. In any case, the

Godot/GDScript framework that was laid down enables quicker creation of more advanced virtual activity algorithms;

- It was clear that lack of reliable haptic interaction measurements represented a major constraint for the experiments presented here. Development of more robust contact force/torque estimation methods, maybe even involving signal fusion with e.g. simplified sEMG sensors, is imperative for this application.

- Instead of manual gain adjustment for elastic actuator velocity control, in order to explicitly compensate for the effects of robot dynamics it might be more efficient to apply the same modeling and identification strategy to the actuator itself in order to match spring interaction and remote reflected forces. That would even be more appropriate for implementations using simpler motor drives which only offer voltage control and offer no reliable velocity maintenance. The extra complexity of a third optimal controller was intentionally avoided (not least because local control was not the main concern for the experiments) but it is an accessible next step;

- For the purpose of this research, the idea from literature that transparent force reflection is necessary for either rehabilitation assistance or gameplay (ATASHZAR; POLUSHIN; PATEL, 2013; SHAHBAZI et al., 2016; MINGE et al., 2017) is readily accepted, but the extent to which that type of interaction is essential was not a subject of discussion or investigation. Perhaps the issue of nonpassive behaviour could be entirely avoided by relying more on local adaptive control algorithms and limiting the remote counterpart to higher-level supervisory commands.

- More advanced resources for social interaction, like VoIP (*voice/video over IP*), and immersion, like VR/AR (*virtual/augmented reality*), also were not experimented with. Those technologies could even improve transparency perception in teleoperation by applying the concept of embodiment (TOET et al., 2020). Apart from developing the additional software infrastructure to integrate those features, it would be interesting to come up with metrics beyond questionnaires to measure their impact on therapy.

- Finally, development and evaluation of lower-cost devices (more affordable than the EXO-TAO) is a relevant effort that could be simultaneously carried out, as it complements the home care enabling purpose of this work (LI et al., 2015; GORSIC; NOVAK, 2016; DARZI; GORŠIČ; NOVAK, 2017; MINGE et al., 2017). In particular, lightweight and structurally flexible designs are interesting, as extra custom springs of SEAs may be replaced by intrinsically elastic links – as it is the case of *EduExo*[1] robotic kit – and nylon thread actuators (HAINES et al., 2014), simplifying transportation and assembly.

[1] https://www.eduexo.com/eduexo-kit/

REFERENCES

ALTENBERG, J. Inside Real-Time Linux. In: EMBEDDED Linux Conference Europe. Berlin: 2016. Available from: <https://www.linux.com/news/event/elce/2017/2/inside-real-time-linux>.

ANDERSON, R. J.; SPONG, M. W. Bilateral Control of Teleoperators with Time Delay. **IEEE Transactions on Automatic Control**, v. 34, n. 5, p. 494–501, 1989. ISSN 15582523. DOI: 10.1109/9.24201.

ANDRADE, K. O. et al. Rehabilitation robotics and serious games: An initial architecture for simultaneous players. **ISSNIP Biosignals and Biorobotics Conference**, 2013. ISSN 23267771. DOI: 10.1109/BRC.2013.6487455.

ATASHZAR, S. F.; POLUSHIN, I. G.; PATEL, R. V. Networked teleoperation with non-passive environment: Application to tele-rehabilitation. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems.: Oct. 2012. p. 5125–5130. ISBN 2153-0858. DOI: 10.1109/IROS.2012.6386152.

_____. Projection-based force reflection algorithms for teleoperated rehabilitation therapy. In: IEEE International Conference on Intelligent Robots and Systems.: 2013. p. 477–482. ISBN 978-1-4673-6358-7. DOI: 10.1109/IROS.2013.6696394.

ATASHZAR, S. F.; SAXENA, A., et al. Involuntary movement during haptics-enabled robotic rehabilitation: Analysis and control design. In: IEEE International Conference on Intelligent Robots and Systems.: 2014. p. 3561–3566. ISBN 978-1-4799-6934-0. DOI: 10.1109/IROS.2014.6943060.

ATASHZAR, S. F.; SHAHBAZI, M., et al. A Passivity-Based Approach for Stable Patient-Robot Interaction in Haptics-Enabled Rehabilitation Systems: Modulated Time-Domain Passivity Control. **IEEE Transactions on Control Systems Technology**, v. 25, n. 3, p. 991–1006, 2017. ISSN 10636536. DOI: 10.1109/TCST.2016.2594584.

BAUR, K.; SCHÄTTIN, A., et al. Trends in robot-assisted and virtual reality-assisted neuromuscular therapy: a systematic review of health-related multiplayer games. **Journal of neuroengineering and rehabilitation**, v. 15, n. 1, p. 107, 2018. ISSN 17430003. DOI: 10.1186/s12984-018-0449-9.

BAUR, K.; WOLF, P., et al. Making neurorehabilitation fun: Multiplayer training via damping forces balancing differences in skill levels. **Proceedings of the 2017 IEEE International Conference on Rehabilitation Robotics**, 2017.

CAO, J. et al. Control strategies for effective robot assisted gait rehabilitation: the state of art and future prospects. **Medical Engineering & Physics**, v. 36, n. 12, p. 1555–66, 2014. ISSN 1873-4030. DOI: 10.1016/j.medengphy.2014.08.005.

CARIGNAN, C. R.; KREBS, H. I. Telerehabilitation robotics: Bright lights, big future? **The Journal of Rehabilitation Research and Development**, v. 43, n. 5, p. 695, 2006. ISSN 0748-7711. DOI: 10.1682/JRRD.2005.05.0085.

CONSONI, L. J. **Leonardo José Consoni Adaptable System for Robotic Telerehabilitation with Serious Games**. 2017. MA thesis – São Carlos School of Engineering - University of São Paulo, available from: <http://www.teses.usp.br/teses/disponiveis/18/18149/tde-15052017-165044/pt-br.php>.

CONSONI, L. J.; PASQUAL, T. B., et al. A robotic telerehabilitation game system for multiplayer activities. In: IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics. Singapore: IEEE, June 2016. 2016-July, p. 798–803. ISBN 978-1-5090-3287-7. DOI: 10.1109/BIOROB.2016.7523725.

CONSONI, L. J.; PEÑA, G. G.; SIQUEIRA, A. A. G. Patient torque estimation for optimal control of a robotic rehabilitation device. In: ENCONTRO Nacional de Engenharia Biomecânica. Águas de Lindóia, Brasil: 2018. p. 4–9.

CONSONI, L. J.; SIQUEIRA, A. A. G.; KREBS, H. I. Compensating for Telecommunication Delays during Robotic Telerehabilitation. In: 15TH IEEE Conference on Rehabilitation Robotics. London: 2017.

COWAN, B.; KAPRALOS, B. A Survey of Frameworks and Game Engines for Serious Game Development. In: 2014 IEEE 14th International Conference on Advanced Learning Technologies.: July 2014. p. 662–664. ISBN 2161-377X. DOI: 10.1109/ICALT.2014.194.

DARZI, A.; GORŠIČ, M.; NOVAK, D. Difficulty adaptation in a competitive arm rehabilitation game using real-time control of arm electromyogram and respiration. In: IEEE International Conference on Rehabilitation Robotics.: 2017. p. 857–862. ISBN 978-1-5386-2296-4. DOI: 10.1109/ICORR.2017.8009356.

DEBRAY, Y. Leveraging Modelica & FMI in Scilab open-source engineering software. In: SCILAB Modelica Conference.: 2015.

DONATI, A. R. C. et al. Long-Term Training with a Brain-Machine Interface-Based Gait Protocol Induces Partial Neurological Recovery in Paraplegic Patients. **Scientific Reports**, v. 6, n. 1, p. 30383, 2016. ISSN 2045-2322. DOI: 10.1038/srep30383. Available from: <https://doi.org/10.1038/srep30383>.

DOS SANTOS, W. M.; CAURIN, G. A.P.; SIQUEIRA, A. A.G. Design and control of an active knee orthosis driven by a rotary Series Elastic Actuator. **Control Engineering Practice**, v. 58, p. 307–318, Jan. 2017. ISSN 0967-0661. DOI: 10.1016/j.conengprac.2015.09.008. Available from: <http://www.sciencedirect.com/science/article/pii/S0967066115300198>.

DOS SANTOS, W. M.; SIQUEIRA, A. A. G. Optimal impedance control for robot-aided rehabilitation of walking based on estimation of patient behavior. In: 2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob).: 2016. p. 1023–1028. ISBN 2155-1782. DOI: 10.1109/BIOROB.2016.7523765.

DURANDAU, G.; FARINA, D.; SARTORI, M. Robust Real-Time Musculoskeletal Modeling Driven by Electromyograms. **IEEE Transactions on Biomedical Engineering**, v. 65, n. 3, p. 556–564, 2018. ISSN 15582531. DOI: 10.1109/TBME.2017.2704085.

FARAGHER, R. **Understanding the basis of the kalman filter via a simple and intuitive derivation [lecture notes]**.: 2012. v. 29. ISBN 1053-5888. DOI: 10.1109/MSP.2012.2203621.

FIEDLER, G. **Networked Physics**.: 2014. Available from: <http://gafferongames.com/networked-physics/introduction-to-networked-physics/>. Visited on: 24 May 2016.

FRIGERIO, M. et al. RobCoGen: a code generator for efficient kinematics and dynamics of articulated robots, based on Domain Specific Languages. **Journal of Software Engineering for Robotics**, v. 7, n. 1, p. 36–54, 2016. ISSN 2035-3928.

FRITZSON, P.; THIELE, Bernhard. **Introduction to Object - Oriented Modeling, Simulation, Debugging and Dynamic Optimization with Modelica using OpenModelica**.: 2016. ISBN 0-471-47163-1.

GONÇALVES, A. C. B. F. et al. Serious games for assessment and rehabilitation of ankle movements. In: IEEE 3rd International Conference on Serious Games and Applications for Health. Rio de Janeiro: IEEE, 2014. p. 1–6. ISBN 978-1-4799-4823-9. DOI: 10.1109/SeGAH.2014.7067071.

GORŠIČ, M.; DARZI, A.; NOVAK, D. Comparison of two difficulty adaptation strategies for competitive arm rehabilitation exercises. **IEEE International Conference on Rehabilitation Robotics**, p. 640–645, 2017. ISSN 19457901. DOI: 10.1109/ICORR.2017.8009320.

GORSIC, M.; NOVAK, D. Design and pilot evaluation of competitive and cooperative exercise games for arm rehabilitation at home. **Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology**

Society, EMBS, 2016-Octob, p. 4690–4694, 2016. ISSN 1557170X. DOI: 10.1109/EMBC.2016.7591774.

HAINES, C. S. et al. Artificial Muscles from Fishing Line and Sewing Thread. **Science**, v. 343, n. 6173, p. 868–872, 2014. ISSN 0036-8075. DOI: 10.1126/science.1246906.

HALL, B. **Beej's Guide to Network Programming**.: Jorgensen Publishing, 2011.

HIRCHE, S.; BUSS, M. Human-oriented control for haptic teleoperation. **Proceedings of the IEEE**, v. 100, n. 3, p. 623–647, 2012. ISSN 00189219. DOI: 10.1109/JPROC.2011.2175150.

HOGAN, N. Impedance Control: An Approach to Manipulation: Part I—Theory. **Journal of Dynamic Systems, Measurement, and Control**, v. 107, n. 1, p. 1, 1985. ISSN 00220434. DOI: 10.1115/1.3140702.

_____. Robot-Aided Neuro-Recovery. **Mechanical Engineering**, v. 136, n. 9, s3–s5, 2014.

HOGAN, N. et al. Motions or muscles? Some behavioral factors underlying robotic assistance of motor recovery. **Journal of rehabilitation research and development**, v. 43, n. 5, p. 605–618, 2006. ISSN 0748-7711. DOI: 10.1682/JRRD.2005.06.0103.

HOKAYEM, P. F.; SPONG, M. W. Bilateral teleoperation: An historical survey. **Automatica**, v. 42, n. 12, p. 2035–2057, 2006. ISSN 00051098. DOI: 10.1016/j.automatica.2006.06.027. Available from: <https://www.sciencedirect.com/science/article/pii/S0005109806002871>.

JARRASSÉ, N.; CHARALAMBOUS, T.; BURDET, E. A Framework to Describe, Analyze and Generate Interactive Motor Behaviors. **PLoS ONE**, v. 7, n. 11, Jan. 2012. ISSN 19326203. DOI: 10.1371/journal.pone.0049945.

JUTINICO, A. L. et al. Impedance control for robotic rehabilitation: A robust markovian approach. **Frontiers in Neurorobotics**, v. 11, AUG, p. 1–16, 2017. ISSN 16625218. DOI: 10.3389/fnbot.2017.00043.

KARIME, A. et al. CAHR: A contextually adaptive home-based rehabilitation framework. **IEEE Transactions on Instrumentation and Measurement**, v. 64, n. 2, p. 427–438, 2015. ISSN 00189456. DOI: 10.1109/TIM.2014.2342432.

KAWAI, Y. et al. Tele-Rehabilitation System for Human Lower Limb using Electrical Stimulation based on Bilateral Teleoperation., p. 1446–1451, 2017. DOI: 10.1109/CCTA.2017.8062662.

KAWASE, K.; MIYOSHI, T.; TERASHIMA, K. Development of multilateral tele-control game using websocket and physics engine. **2015 IEEE/SICE International Symposium on System Integration, SII 2015**, p. 265–270, 2016. DOI: 10.1109/SII.2015.7404989.

KREBS, H. I. et al. A paradigm shift for rehabilitation robotics. **IEEE Engineering in Medicine and Biology Magazine**, v. 27, n. 4, p. 61–70, 2008. ISSN 07395175. DOI: 10.1109/MEMB.2008.919498.

LANINI, J. et al. Teleoperation of two six-degree-of-freedom arm rehabilitation exoskeletons. In: IEEE International Conference on Rehabilitation Robotics. Singapore: 2015. p. 514–519. ISBN 978-1-4799-1807-2. DOI: 10.1109/ICORR.2015.7281251.

LARSEN, T. D. et al. Incorporation of time delayed measurements in a discrete-time Kalman filter., p. 3972–3977, 2002. DOI: 10.1109/cdc.1998.761918.

LI, S. et al. A Mobile Cloud Computing Framework Integrating Multilevel Encoding for Performance Monitoring in Telerehabilitation., v. 2015, 2015.

AL-MAHMOOD, A.; AGYEMAN, M. O. On wearable devices for motivating patients with upper limb disability via gaming and home rehabilitation. **2018 3rd International Conference on Fog and Mobile Edge Computing, FMEC 2018**, p. 155–162, 2018. DOI: 10.1109/FMEC.2018.8364058.

MARCHAL-CRESPO, L.; REINKENSMEYER, D. J. Review of control strategies for robotic movement training after neurologic injury. **Journal of Neuroengineering and Rehabilitation**, v. 6, n. 1, p. 20, 2009. ISSN 1743-0003. DOI: 10.1186/1743-0003-6-20.

MATSANGIDOU, M. et al. Is your virtual self as sensational as your real? Virtual Reality: The effect of body consciousness on the experience of exercise sensations. **Psychology of Sport and Exercise**, v. 41, p. 218–224, Mar. 2019. ISSN 1469-0292. DOI: 10.1016/J.PSYCHSPORT.2018.07.004. Available from: <https://www.sciencedirect.com/science/article/pii/S1469029217306246?via=ihub>.

MEFOUED, S.; MOHAMMED, S.; AMIRAT, Y. Toward Movement Restoration of Knee Joint Using Robust Control of Powered Orthosis., v. 21, n. 6, p. 1–13, 2013.

MINGE, M. et al. BeMobil: Developing a User-Friendly and Motivating Telerehabilitation System for Motor Relearning after Stroke*. In: ISBN 978-1-5386-2295-7.

MIYOSHI, T.; TERASIMA, K.; BUSS, M. A design method of wave filter for stabilizing non-passive operating system. **Proceedings of the IEEE International Conference on Control Applications**, p. 1318–1324, 2006. DOI: 10.1109/CACSD-CCA-ISIC.2006.4776833.

MUNIR, S.; BOOK, W. J. Internet-based teleoperation using wave variables with prediction. **IEEE/ASME Transactions on Mechatronics**, v. 7, n. 2, p. 124–133, 2002. ISSN 10834435. DOI: 10.1109/TMECH.2002.1011249.

NIEMEYER, G.; SLOTINE, J.-J. E. Telemanipulation with Time Delays. **The International Journal of Robotics Research**, v. 23, n. 9, p. 873–890, 2004. ISSN 0278-3649. DOI: 10.1177/0278364904045563.

NORMAN, S. L.; LOBO-PRAT, J.; REINKENSMEYER, D. J. How do strength and coordination recovery interact after stroke? A computational model for informing robotic training. **IEEE International Conference on Rehabilitation Robotics**, p. 181–186, 2017. ISSN 19457901. DOI: 10.1109/ICORR.2017.8009243.

NOVAK, D. et al. Increasing motivation in robot-aided arm rehabilitation with competitive and cooperative gameplay. **Journal of neuroengineering and rehabilitation**, v. 11, p. 64, 2014. ISSN 1743-0003. DOI: 10.1186/1743-0003-11-64.

PEÑA, G. G. et al. Feasibility of an optimal EMG-driven adaptive impedance control applied to an active knee orthosis. **Robotics and Autonomous Systems**, v. 112, p. 98–108, 2019. ISSN 09218890. DOI: 10.1016/j.robot.2018.11.011. Available from: <https://www.sciencedirect.com/science/article/pii/S0921889018304263?via%3Dihub>.

PEREZ-IBARRA, J. C.; SIQUEIRA, A. A. G.; KREBS, H. I. Assist-As-needed ankle rehabilitation based on adaptive impedance control. In: IEEE International Conference on Rehabilitation Robotics.: 2015. 2015-September, p. 723–728. ISBN 978-1-4799-1807-2. DOI: 10.1109/ICORR.2015.7281287.

PÉREZ-IBARRA, J. C. et al. Adaptive Impedance Control Applied to Robot-Aided Neuro-Rehabilitation of the Ankle. **IEEE Robotics and Automation Letters**, v. 4, n. 2, p. 185–192, Apr. 2019. ISSN 2377-3774. DOI: 10.1109/LRA.2018.2885165.

PIZZOLATO, C. et al. Real-time inverse kinematics and inverse dynamics for lower limb applications using OpenSim. **Computer Methods in Biomechanics and Biomedical Engineering**, November, p. 1–10, 2016. ISSN 1025-5842. DOI: 10.1080/10255842.2016.1240789.

PRAHM, C. et al. Increasing motivation, effort and performance through game-based rehabilitation for upper limb myoelectric prosthesis control. **International Conference on Virtual Rehabilitation, ICVR**, 2017-June, 2017. ISSN 23319569. DOI: 10.1109/ICVR.2017.8007517.

RODRÍGUEZ-SEDA, E. J. Passive Transparency Compensation for Bilateral Teleoperators with Communication Delays., v. 2015, 2015.

RODRÍGUEZ-SEDA, E. J.; LEE, D.; SPONG, M. W. Experimental comparison study of control architectures for bilateral teleoperators. **IEEE Transactions on Robotics**, v. 25, n. 6, p. 1304–1318, 2009. ISSN 15523098. DOI: 10.1109/TRO.2009.2032964.

ROY, A. et al. Robot-Aided Neurorehabilitation: A Novel Robot for Ankle Rehabilitation. **IEEE Trans Robotics**, v. 25, n. 3, p. 569–582, 2009. ISSN 1552-3098. DOI: 10.1109/TRO.2009.2019783.

SANTOS, W. M. et al. Design and Evaluation of a Modular Lower Limb Exoskeleton for Rehabilitation *., p. 447–451, 2017. ISSN 9781538622964. DOI: 10.1109/ICORR.2017.8009288.

SARTORI, M.; FARINA, D. Neural Data-driven Musculoskeletal Modeling for Neurorehabilitation Technologies. **IEEE Transactions on Biomedical Engineering**, v. 63, n. 5, p. 879–893, 2016. ISSN 0018-9294. DOI: 10.1109/TBME.2016.2538296.

SETH, A. et al. OpenSim: a musculoskeletal modeling and simulation framework for in silico investigations and exchange. **Procedia IUTAM**, v. 2, p. 212–232, Jan. 2011. ISSN 2210-9838. DOI: 10.1016/j.piutam.2011.04.021.

SHAHBAZI, M. et al. Robotics-assisted mirror rehabilitation therapy: A therapist-in-the-loop assist-as-needed architecture. **IEEE/ASME Transactions on Mechatronics**, v. 21, n. 4, p. 1954–1965, 2016. ISSN 10834435. DOI: 10.1109/TMECH.2016.2551725.

SKOGESTAD, S.; POSTLETHWAITE, I. **Multivariable feedback control: analysis and design**.: 2005. v. 8. ISBN 978-0-470-01168-3.

SUZNJEVIC, M.; SALDANA, J. **Delay Limits for Real-Time Services**.: 2015. Available from: <https://tools.ietf.org/id/draft-suznjevic-dispatch-delay-limits-00.html>. Visited on: 20 Jan. 2017.

TAGLIAPIETRA, L. et al. Estimating EMG signals to drive neuromusculoskeletal models in cyclic rehabilitation movements. **Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS**, 2015-Novem, p. 3611–3614, 2015. ISSN 1557170X. DOI: 10.1109/EMBC.2015.7319174.

TODOROV, E.; LI, W. A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. **Proceedings of the 2005, American Control Conference, 2005.**, p. 300–306, 2005. ISSN 0743-1619. DOI: 10.1109/ACC.2005.1469949. Available from: <http://ieeexplore.ieee.org/document/1469949/>.

TOET, A. et al. Toward Enhanced Teleoperation Through Embodiment. **Frontiers in Robotics and AI**, v. 7, p. 14, 2020. ISSN 2296-9144. DOI: 10.3389/frobt.2020.00014. Available from: <https://www.frontiersin.org/article/10.3389/frobt.2020.00014>.

VIVIAN, M.; REGGIANI, M.; SARTORI, M. Experimentally-based optimization of contact parameters in dynamics simulation of humanoid robots. In: IEEE International Conference on Robotics and Automation.: 2013. p. 1643–1648. ISBN 978-1-4673-5641-1. DOI: 10.1109/ICRA.2013.6630790.

WENK, N. et al. Reaching in Several Realities: Motor and Cognitive Benefits of Different Visualization Technologies., p. 1037–1042, 2019. DOI: 10.1109/icorr.2019.8779366.

WILDENBEEST, J. G.W.; ABBINK, D. A.; SCHORSCH, J. F. Haptic transparency increases the generalizability of motor learning during telemanipulation. In: 2013 World Haptics Conference (WHC).: Apr. 2013. p. 707–712. ISSN: null. DOI: 10.1109/WHC.2013.6548495.

WORLD HEALTH ORGANIZATION. **Global burden of stroke**.: 2004. v. 15. ISBN 92-4-156276-5. DOI: 10.1016/B978-1-4160-5478-8.10019-3.

_____. International Perspectives on Spinal Cord Injury. **The Lancet**, p. 4, 2013.

WYETH, G. Control issues for velocity sourced series elastic actuators., Jan. 2006.

Appendix

APPENDIX A – ROBOT CONTROL MESSAGING FORMAT

When working with inter-computer communication involving explicit memory buffer copies, like in *RobotControl*, an issue to be taken into consideration is the byte ordering for numeric values (integer and floating point) used in each system. Some hardware architectures like *x86* use *Little-Endian*, storing least and most significant bytes in the reverse memory order of systems employing *Big-Endian* (HALL, 2011).

In practice that issue has little impact, since two of the predominant architectures, namely *x86* and ARM, support *Little-Endian* format. By respecting that restriction and complying with other encoding conventions like IEEE 754 for floating point (guaranteed for *ISO C99* compliant compilers), there is no need for data conversions overhead.

A.1 Request/Reply TCP Message Opcodes

- **LIST CONFIGS** (0x01): request information about available robot configurations. Its reply is followed by a serialized JSON (customizable) string such as:

```
{"robots":["robot_1","robot_2","robot_3"]}
```

Code A.1 – Example of JSON-formatted list of available configurations

- **GET INFO** (0x02): request information about all available robot's axes and joints. Its reply is followed by a serialized JSON (customizable) string such as:

```
{"id":"robot_1","axes":["r1_x","r1_y"],"joints":["r1_0","r1_1"]}
```

Code A.2 – Example of JSON-formatted information for a robot with 2 DoFs

- **SET USER** (0x03): set user name for data logging. Followed by the user string
- **SET CONFIG** (0x04): set a new robot configuration (reload JSON files). Is followed by the robot name string
- **DISABLE** (0x05): turn actuators off and disable control
- **ENABLE** (0x06): turn actuators on and make control active
- **PASSIVATE** (0x07): set the robot control to passive/compliant behavior
- **OFFSET** (0x08): set the robot control to joints reference acquisition
- **CALIBRATE** (0x09): set the robot control to joints amplitude measurement
- **OPERATE** (0x0A): set the robot control to normal operation

A.2 Data Update UDP Message Array Format

Table 10 – Message format for axes updates (input or output) exchange with *RobotControl*.

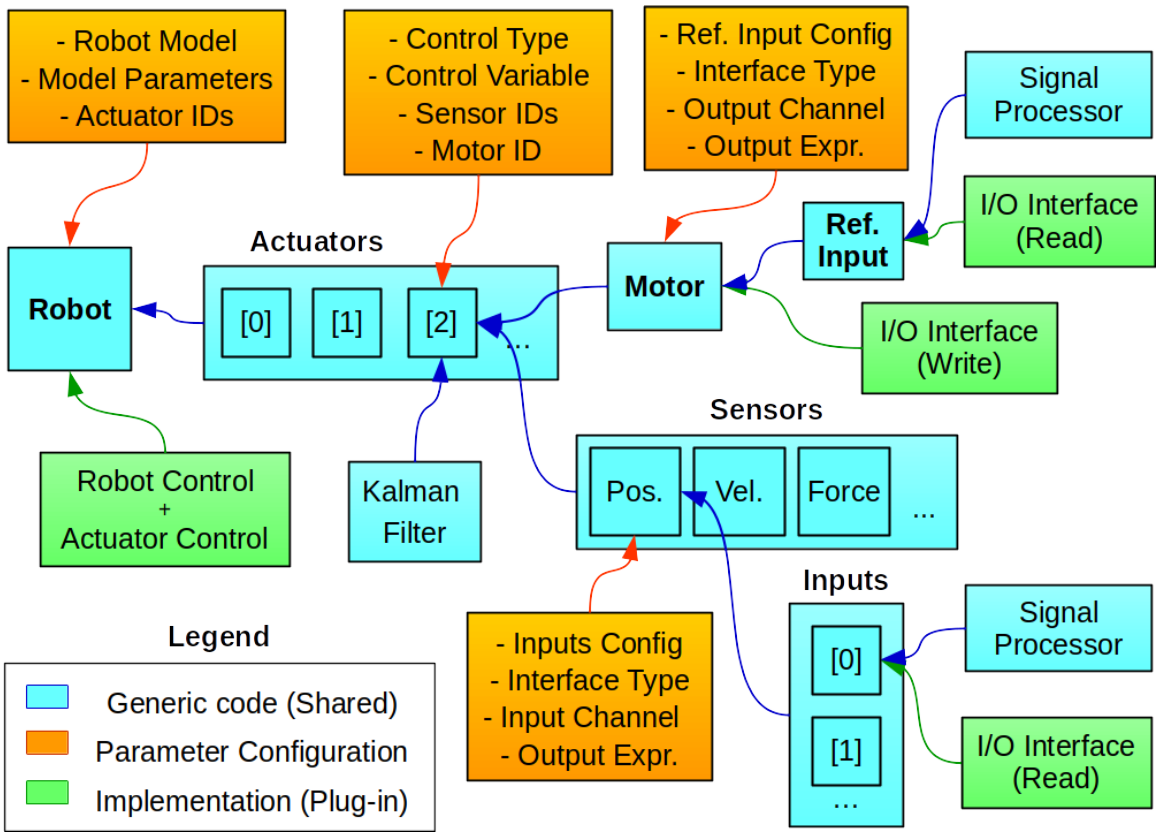
1 byte	1 b	4 b	4 b	4 b	4 b	4 b	4 b	4 b	1 b	...
nº DoFs	DoF index 1	pos.	vel.	accel.	force	inert.	damp.	stiff.	DoF index 2	...

Source: The author

APPENDIX B – ROBOT CONFIGURATION LAYERS

The *RobotControl* device configuration scheme presented in Fig. 57 (and in online documentation[1].) is a composition of 4 abstraction layers intended to generalize (light blue boxes) the structure of any given robot (not limited to rehabilitation).

Figure 57 – Diagram of *RobotControl* implementation abstraction levels.



Source: The author, modified from Consoni (2017)

The hierarchical tree of robot, actuators, sensors/motors and inputs/outputs is specified via a set of text files (orange boxes) that reference each other, which is also used for common parameter options (Appendix D). Green boxes represent the developer provided plug-ins (Appendix C) implementing specific control logic and coordinate conversions for each robot or I/O hardware (e.g. signal acquisition boards, motor drives) communication.

In generic layers made of lists of components (displayed by identified squares inside the blue rectangles)the subcomponents, parameter files and implementation modules pointing to one box actually have to be provided for all elements of that set.

[1] https://github.com/EESC-MKGroup/RobotSystem-Lite#robot-multi-level-configuration

From top to bottom levels, the full control application is composed of:

- The base robot instance (only 1 per control process), configured according to a provided list of identifiers (IDs), with each object defined by:
  - Robot configuration parameters
  - Robot-actuator control implementation module
  - A list of generic actuators list, each one, in turn, complemented by:
    - Actuator configuration parameters
    - A *Kalman* filter for sensor fusion (FARAGHER, 2012)
    - A list of generic sensors, each one, in turn, completed by:
      - Sensor configuration parameters
      - Data acquisition/inputs implementation module
    - Generic actuation motor, complemented by:
      - Motor configuration parameters
      - Signal output implementation module

APPENDIX C – PLUG-IN INTERFACES DESCRIPTION

Sensor and motor boards I/O and robot control algorithms are procedures which present too much variation – from one kind of device to another – to be simply parametrized. Having optimized code for those operations seems more efficient.

To enable such feature while keeping its generality, *RobotControl* supports dynamic loading of binary libraries (like DLLs on *Windows* OS or Shared Objects on *Unix*). By following defined plug-in APIs, one could write and build its own device-specific implementation module to be loaded and called by the core control application at runtime.

C.1 Signal I/O Interface

Physical/virtual signal aquisition and generation interface is described below. More detailed documentation can be found at the header repository[1]:

```
/// Creates plugin specific signal I/O device data structure
long InitDevice( const char* deviceConfig );
/// Discards given signal I/O device data structure
void EndDevice( long deviceID );
/// Verifies occurence of errors on given device
bool HasError( long deviceID );
/// Resets data and errors for given device
void Reset( long deviceID );
/// Gets number of samples aquired for every given device input
   channel on each Read() call
size_t GetMaxInputSamplesNumber( long deviceID );
/// Check reading availability for device's input channel
bool CheckInputChannel( long deviceID, unsigned int channel );
/// Reads samples list from device's specified channel
size_t Read( long deviceID, unsigned int channel,
             double* ref_value );
/// Get exclusive access to device's output channel
bool AcquireOutputChannel( long deviceID, unsigned int channel );
/// Give up exclusive access to device's output channel
void ReleaseOutputChannel( long deviceID, unsigned int channel );
/// Writes value to specified channel of given device
bool Write( long deviceID, unsigned int channel, double value );
```

Code C.1 – File/string/stream data input/output methods to be implemented by plugins.

[1] https://github.com/EESC-MKGroup/Signal-IO-Interface

C.2 Robot Control Interface

Common robot control interface is described below. More detailed documentation can be found on the header repository[2]:

```
/// Calls plugin specific robot controller initialization
bool InitController( const char* configurationString );
/// Calls plugin specific robot controller data deallocation
void EndController( void );
/// Calls plugin specific logic to process single control pass
/// and joints/axes coordinate conversions
void RunControlStep( DoFVariables** jointMeasuresList,
                     DoFVariables** axisMeasuresList,
                     DoFVariables** jointSetpointsList,
                     DoFVariables** axisSetpointsList,
                     double timeDelta );
/// Trigger control state change for plugin specific behaviour
void SetControlState( enum ControlState controlState );
/// Get plugin specific number of joint coordinates
size_t GetJointsNumber( void );
/// Get plugin specific names of all joints
const char** GetJointNamesList( void );
/// Get plugin specific number of axis coordinates
size_t GetAxesNumber( void );
/// Get plugin specific names of all axes
const char** GetAxisNamesList( void );
/// Get number of additional inputs needed for the robot control
size_t GetExtraInputsNumber( void );
/// Set list of additional inputs for the next control step
void SetExtraInputsList( double* inputsList );
/// Get number of additional outputs provided by controller
size_t GetExtraOutputsNumber( void );
/// Get list of additional outputs from the last control step
void GetExtraOutputsList( double* outputsList );
```

Code C.2 – Robot control methods to be implemented by plugins.

[2] https://github.com/EESC-MKGroup/Robot-Control-Interface

APPENDIX D – CONFIGURATION FILES OPTIONS

This is a complete list of *RobotControl* available parameters for its multi-level robot configuration system (Appendix B). In the default parsing implementation, configuration files are written in JSON (*JavaScript Object Notation*) format[1], loaded at execution time.

The JSON file parser itself is implemented as a backend for a generic Data I/O interface[2], making possible transitions to other storage formats, like XML (*Extensible Markup Language*) files or SQL (*Structured Query Language*) databases.

D.1 Robot-level

Below are robot layer settings (entries marked with 'o' are optional and display their default values). More detailed documentation can be found online[3]:

```
{
  "controller": {                // Robot controller configuration
  "type": "<library_name>",      // Plug-in library path
  "config": "",                  // [o] Custom configuration string
  "time_step": 0.005             // [o] Control time step
  },
  "actuators": [                 // List of robot's actuators
    "<actuator_1_id>",           // Actuator string identifier
    "<actuator_2_id>", ...
  ],
  "extra_inputs": [              // [o] Additional control inputs
    { "interface": { ... },
      "signal_processing": { ... } }, ...
  ],
  "extra_outputs": [             // [o] Additional control outputs
    { "interface": { ... } }, ...
  ]
  "log": {                       // [o] Set logging properties
    "to_file": false,            // [o] Save data to file
    "precision": 3               // [o] Decimal values precision
  }
}
```

Code D.1 – Possible parameters for robot-level configuration.

[1] http://www.json.org/
[2] https://github.com/EESC-MKGroup/Data-IO-Interface
[3] https://eesc-mkgroup.github.io/RobotSystem-Lite/robot_config.html

D.2 Actuator-level

Below are actuator layer settings (entries marked with 'o' are optional and display their default values). More detailed documentation can be found online[4].

```
{
  "sensors": [                          // List of used sensors
    {
      "variable": "POSITION",           // Measured variable
      "config": "<sensor_1_id>",        // Sensor string identifier
      "deviation": 1.0                  // [o] Measurement error
    },
    {
      "variable": "FORCE",
      "config": "<sensor_2_id>"
    }, ...
  ],
  "motor": {                            // Actuation motor
    "variable": "VELOCITY",             // Controlled variable
    "config": "<motor_identifier>",     // Motor string identifier
    "limit": -1.0                       // [o] Absolute max output
  },
  "log": {                      // [o] Set logging properties
    "to_file": false,           // [o] Save data to file
    "precision": 3              // [o] Decimal values precision
  }
}
```

Code D.2 – Possible parameters for actuator-level configuration.

[4] https://eesc-mkgroup.github.io/RobotSystem-Lite/actuator_config.html

D.3 Sensor-level

Below are sensor layer settings (entries marked with 'o' are optional and display their default values). More detailed documentation can be found online[5].

```
{
  "inputs": [                    // List with input sources
   {
      "interface": {                 // I/O interface properties
        "type": "<library_name>",    // Plug-in library path
        "config": "",                // [o] Custom config. string
        "channel": 0                 // Device input channel
      },
      "signal_processing": {     // [o] Signal processing options
        "rectified": false,      // [o] Rectify signal if true
        "normalized": false,     // [o] Normalize signal if true
        "min_frequency": -1.0    // [o] Low-pass IIR filter freq.
        "max_frequency": -1.0    // [o] High-pass IIR filter freq.
      }
    }, ...
  ],
  "output": "in0",            // [o] Input-output conversion expr.
  "log": {                    // [o] Set logging properties
    "to_file": false,         // [o] Save data to file
    "precision": 3            // [o] Decimal values precision
  }
}
```

Code D.3 – Possible parameters for sensor-level configuration.

[5] https://eesc-mkgroup.github.io/RobotSystem-Lite/sensor_config.html

D.4 Motor-level

Below are motor layer settings (entries marked with 'o' are optional and display their default values). More detailed documentation can be found online[6].

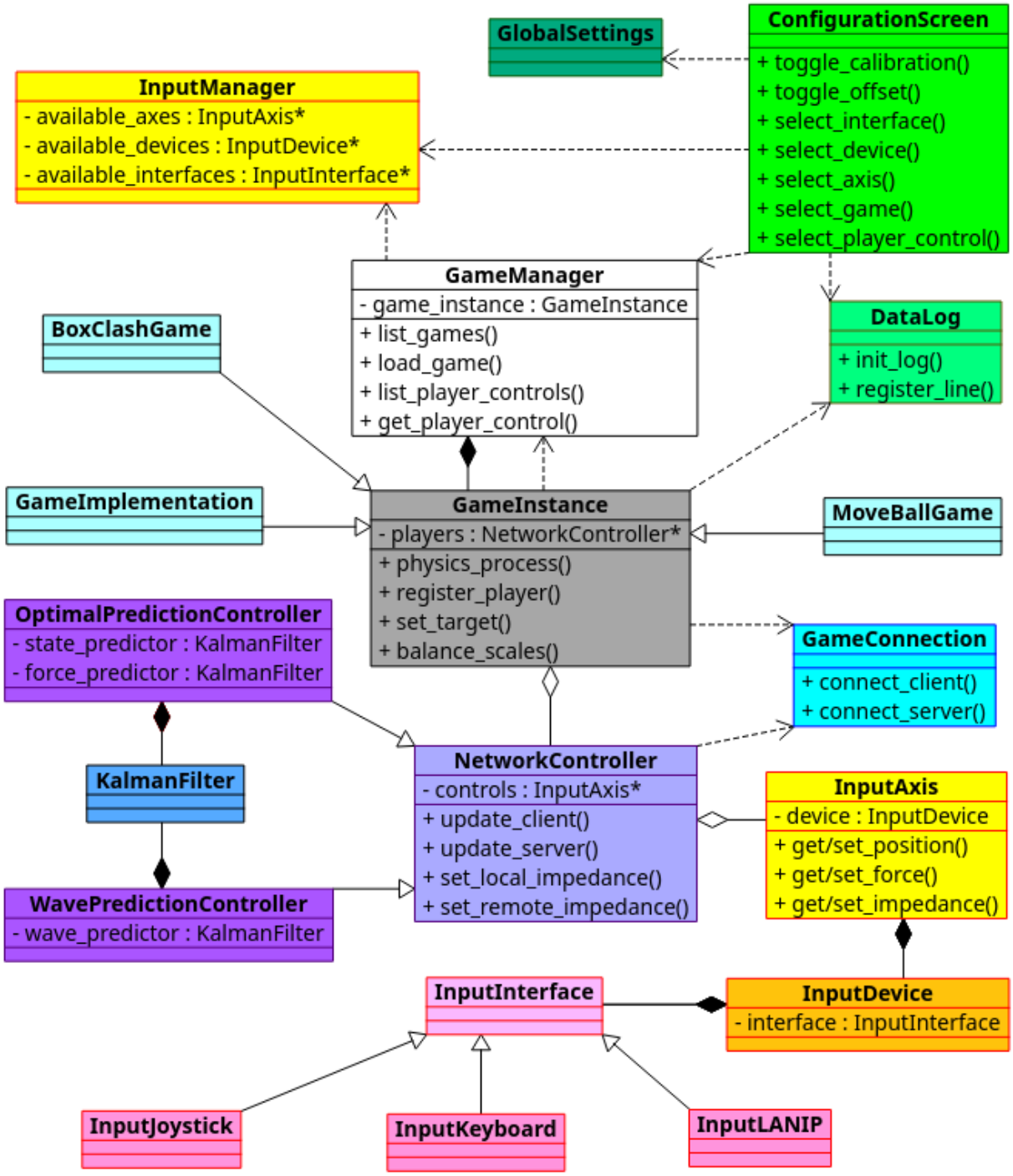
```
{
  "interface": {                // I/O interface properties
    "type": "<library_name>",   // Plug-in library path
    "config": "",               // [o] Custom config. string
    "channel": 0                // Device output channel
  },
  "reference": {                // [o] Offset ref. input
    "interface": { ... },
    "signal_processing": { ... }
  },
  "output": "set",            // [o] Input-output conversion expr.
  "log": {                    // [o] Set logging properties
    "to_file": false,         // [o] Save data to file
    "precision": 3            // [o] Decimal values precision
  }
}
```

Code D.4 – Possible parameters for motor-level configuration.

6 https://eesc-mkgroup.github.io/RobotSystem-Lite/motor_config.html

APPENDIX E – MULTIPLAYER GAME FRAMEWORK MODEL

Figure 58 – Simplified UML diagram of the developed game framework's main components.



Source: The author

The developed *Godot* multiplayer game framework[1] (Fig. 58) is organized as follows: *ConfigurationScreen* contains the methods for startup configuration, axis selection, calibration, etc; *GameInstance* is the base interface that any multiplayer game (e.g. *Box Clash* and *Move Ball*) shall implement; *NetworkController* is the base class on top of that the teleoperators are written; *InputInterface* is the common set of functions through which implementations of *GameInstance* or *NetworkController* access any device's axes.

[1] https://github.com/EESC-MKGroup/Godot-Rehab-Game-Framework

APPENDIX F – PUBLISHED WORKS

The conference and journal articles published or submitted over the course of research period are listed below:

- CONSONI, L. J.; SIQUEIRA, A. A. G.; KREBS, H. I. Compensating for Telecommunication Delays during Robotic Telerehabilitation. In: 15TH IEEE Conference on Rehabilitation Robotics. London: 2017
- CONSONI, L. J.; PEÑA, G. G.; SIQUEIRA, A. A. G. Patient torque estimation for optimal control of a robotic rehabilitation device. In: ENCONTRO Nacional de Engenharia Biomecânica. Águas de Lindóia, Brasil: 2018. p. 4–9
- PEÑA, G. G. et al. Feasibility of an optimal EMG-driven adaptive impedance control applied to an active knee orthosis. **Robotics and Autonomous Systems**, v. 112, p. 98–108, 2019. ISSN 09218890. DOI: 10.1016/j.robot.2018.11.011. Available from: <https://www.sciencedirect.com/science/article/pii/S0921889018304263?via%3Dihub>
- Consoni and Siqueira (2020) [submitted]