Lecture 10: Teleoperation (transparency and stability) and Graphics

Allison M. Okamura
Stanford University
teleoperator
transparency
primary teleoperation performance metrics

tracking
the ability of the follower to follow the master

transparency
(for bilateral teleoperation only)
many definitions, but a popular one is whether the mechanical impedance felt by the user is the same as the impedance of the environment
Z-width for teleoperators

![Graph showing Z-width for teleoperators with impedance and frequency axes.](image)
discussion

• what factors might affect tracking?

• what factors might affect transparency?
system structure

$m = \text{master}, \ s = \text{slave/follower} \ (\text{for consistency with literature})
transparency

requirement for perfect transparency
(impedance reflection):

\[ Z_e(s) = \frac{F_e(s)}{X_s(s)} \]

\[ Z_{felt}(s) = \frac{F_h(s)}{X_m(s)} \]

\[ Z_e(s) = Z_{felt}(s) \]

a more strict requirement would be:

\[ F_h(s) = F_e(s) \quad \text{and} \quad X_m(s) = X_s(s) \]
transparency

are our three controllers transparent?
you can test each one, using:

\[ F_h = X_m Z_m + F_{am} \]
\[ F_e = -X_s Z_s + F_{as} \]

what assumptions are needed to achieve perfect transparency?
This gives the following relationships, but they are challenging because we don’t actually know many of these parameters:

\[ F_h = F_h^* - Z_h V_h, \quad F_e = F_e^* + Z_e V_e \]
transparency using this notation

we want impedance matching:

\[ Z_{to} = Z_e \quad \text{or} \quad Z_{te} = Z_h \]

and perhaps kinematic correspondence as well:

\[ V_h \equiv V_e \]
two-port network model

there are four types
here is one (a hybrid model):

\[
\begin{bmatrix}
F_h \\
-V_e
\end{bmatrix}
:=
\begin{bmatrix}
h_{11} & h_{12} \\
h_{21} & h_{22}
\end{bmatrix}
\begin{bmatrix}
V_h \\
F_e
\end{bmatrix}
\]

the hybrid parameters \( h_{ij} \) are functions of the master and follower dynamics and their control parameters

the transmitted impedances can be computed:

\[
Z_{te} := \left. \frac{F_e}{-V_e} \right|_{F_e^* = 0} = \frac{h_{11} + Z_h}{(h_{11} h_{22} - h_{12} h_{21}) + h_{22} Z_h}
\]

\[
Z_{to} := \left. \frac{F_h}{V_h} \right|_{F_e^* = 0} = \frac{h_{11} + (h_{11} h_{22} - h_{12} h_{21}) Z_e}{1 + h_{22} Z_e}
\]
the transmitted impedances can be computed:

\[ Z_{to} := \frac{F_h}{V_h} \bigg|_{F_e^* = 0} = \frac{h_{11} + (h_{11} h_{22} - h_{12} h_{21}) Z_e}{1 + h_{22} Z_e} \]

\[ Z_{te} := \frac{F_e}{-V_e} \bigg|_{F_e^* = 0} = \frac{h_{11} + Z_h}{(h_{11} h_{22} - h_{12} h_{21}) + h_{22} Z_h} \]

but we don’t know \( Z_e \) and \( Z_h \)!

so impedance matching can only be guaranteed when

\[ h_{11} = h_{22} = 0 \]

\[ h_{12} h_{21} = -1 \]

this will enforce (no matter what \( Z_e \) and \( Z_h \) are)

\[ Z_{to} = Z_e \quad \text{and} \quad Z_{te} = Z_h \]
kinematic correspondence

assuming no position or force scaling, the velocity of the human and the environment are

\[
V_h = \frac{(F_h^* - h_{12}F_e^*) (h_{22}Z_e + 1) + h_{12}h_{22}Z_e F_e^*}{(h_{11} + Z_h) (h_{22}Z_e + 1) - h_{12}h_{21}Z_e}
\]

\[
V_e = -\frac{(h_{11} + Z_h) h_{22}F_e^* + h_{21} (F_h^* - h_{12}F_e^*)}{(h_{11} + Z_h) (h_{22}Z_e + 1) - h_{12}h_{21}Z_e}
\]

this also requires certain \( h \) parameters

\[
h_{12} (h_{21} + 1) = h_{22} (h_{11} + Z_h)
\]

\[
(h_{21} + 1) = -h_{22}Z_e
\]
total transparency

to enforce perfect impedance reflection and kinematic correspondence

\[ Z_{to} = Z_e \quad Z_{te} = Z_h \]
\[ V_h \equiv V_e \]

we must have:

\[ h_{11} = h_{22} = 0 \]
\[ h_{12} = -h_{21} = 1 \]

but what are these \( h \)'s??

Hashtrudi-Zaad & Salcudean, 2002
We are ignoring time delay blocks in this class. If you want to include them, use the Padé approximation you learned in ENGR 105:

\[ e^{-T_d s} \approx \frac{1 - \left( \frac{T_d s}{2} \right)}{1 + \left( \frac{T_d s}{2} \right)} \]
closed-loop equations

master:

\[ Z_{cm} V_h + C_4 e^{-Ts} V_e = (1 + C_6) F_h - C_2 e^{-Ts} F_e \]

follower:

\[ C_1 e^{-Ts} V_h - Z_{cs} V_e = -C_3 e^{-Ts} F_h + (1 + C_5) F_e \]

where:

\[ Z_{cm} = C_m + Z_m \quad Z_{cs} = C_s + Z_s \]
transmitted impedances

\[ Z_{te} = \frac{[Z_{cm} Z_{cs} + C_1 C_4] + [(1 + C_6) Z_{cs} - C_3 C_4] Z_h}{[(1 + C_5) Z_{cm} + C_1 C_2] + [(1 + C_5)(1 + C_6) - C_2 C_3] Z_h} \]

\[ Z_{to} = \frac{[Z_{cm} Z_{cs} + C_1 C_4] + [(1 + C_5) Z_{cm} + C_1 C_2] Z_e}{[(1 + C_6) Z_{cs} - C_3 C_4] + [(1 + C_5)(1 + C_6) - C_2 C_3] Z_e} \]

you can match these equations to the equations with the \( h \)'s to see the relationship between the \( h \)'s and the control parameters of the system
transparency condition

If $C_1 \ldots C_6$ are not functions of $Z_h$ and $Z_e$

and $(C_2, C_3) \neq (0,0)$

\begin{align*}
C_1 &= Z_{cs} \\
C_2 &= 1 + C_6 \\
C_3 &= 1 + C_5 \\
C_4 &= -Z_{cm}
\end{align*}

physical interpretation: the master and follower dynamics have to be cancelled out by inverse dynamics and the feedforward forces have to match the net forces exerted by the operator on the environment

Hashtrudi-Zaad & Salcudean, 2001
teleoperator

stability
two-port network model

there are four types (here are all of them):

\[
\begin{bmatrix}
F_h \\
F_e
\end{bmatrix} = \mathcal{O}_Z := Z I Z = \begin{bmatrix}
z_{11} & z_{12} \\
z_{21} & z_{22}
\end{bmatrix} \begin{bmatrix}
V_h \\
-V_e
\end{bmatrix}
\]

\[
\begin{bmatrix}
V_h \\
-V_e
\end{bmatrix} = \mathcal{O}_Y := Y I Y = \begin{bmatrix}
y_{11} & y_{12} \\
y_{21} & y_{22}
\end{bmatrix} \begin{bmatrix}
F_h \\
F_e
\end{bmatrix}
\]

\[
\begin{bmatrix}
F_h \\
-V_e
\end{bmatrix} = \mathcal{O}_H := H I H = \begin{bmatrix}
h_{11} & h_{12} \\
h_{21} & h_{22}
\end{bmatrix} \begin{bmatrix}
V_h \\
F_e
\end{bmatrix}
\]

\[
\begin{bmatrix}
V_h \\
F_e
\end{bmatrix} = \mathcal{O}_G := G I G = \begin{bmatrix}
g_{11} & g_{12} \\
g_{21} & g_{22}
\end{bmatrix} \begin{bmatrix}
F_h \\
-V_e
\end{bmatrix}
\]

where the \textit{immitance} parameters (called \(p_{ij}\) in general) are functions of the master and follower dynamics and their control parameters.
stability criteria

an LTI two-port network is absolutely stable if and only if

1. $p_{11}$ and $p_{22}$ have no poles in the right half plane

2. any poles of $p_{11}$ and $p_{22}$ on the imaginary axis are simple (of order one) and have real and positive residues. (The residue is the constant $a_{n-1}$ in the series $f(z) = \sum_{x=-\infty}^{\infty} a_x (z - z_0)^x$)

and...
stability criteria

3. the following inequalities hold:

\[ \Re \{ p_{11} \} \geq 0 \]

\[ \Re \{ p_{22} \} \geq 0 \]

\[ 2\Re \{ p_{11} \} \Re \{ p_{22} \} - \Re \{ p_{12} p_{21} \} - |p_{12} p_{21}| \geq 0 \]

on the \( j\omega \) axis for all \( \omega \geq 0 \)
what can you do with this?

you can link up an unconditionally stable system with other passive systems, and the whole thing will be stable

assumptions:
- system is LTI
- no sample and hold, no ZOH
- no time delay or discretization

robustness

“Although four-channel control architectures can provide stable perfectly transparent systems in theory, stability and performance for these systems are compromised due to the communication-channel delay as well as the operator and environment dynamic uncertainties.”

Hashtrudi-Zaad & Salcudean, 2001
Unilateral Teleoperation Setup
implementation summary

**follower robot controller**

\[ f_{a2}(t) = k_{p2}(x_1 - x_2) + k_{d2}(\dot{x}_1 - \dot{x}_2) \]

**master robot controller**

unilateral teleoperation:

\[ f_{a1}(t) = 0 \]

bilateral teleoperation (position-exchange):

\[ f_{a1}(t) = k_{p1}(x_2 - x_1) + k_{d1}(\dot{x}_2 - \dot{x}_1) \]

bilateral teleoperation (position forward, force feedback):

\[ f_{a1}(t) = f_e \]
Hints/Suggestions

- Connect both motors to one Hapkit Board. Call the Hapkit with this board the “master”.
- Connect the MR sensor on the “follower” Hapkit Board to an analog input on the “master”.
- Duplicate all functions in code to include “follower” Hapkit (sections previously do not edit)
- The “follower” MR sensor still needs power!
- Add a common ground between Hapkits!
<table>
<thead>
<tr>
<th>ATmega 328 chip pin #</th>
<th>ATmega 328 pin name</th>
<th>Typical Arduino function</th>
<th>Special Hapkit function</th>
<th>Pin name printed on Hapkit Board</th>
<th>Pin number to use in Arduino program</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PC6 (PCINT14/Reset)</td>
<td>Reset</td>
<td>Reset</td>
<td>RST</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>PD0 (PCINT16/RXD)</td>
<td>Digital Pin 0 (RX)</td>
<td>D0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>PD1 (PCINT17/TXD)</td>
<td>Digital Pin 1 (TX)</td>
<td>D1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>PD2 (PCINT18/INT0)</td>
<td>Digital Pin 2</td>
<td>D2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>PD3 (PCINT19/OC2B/INT1)</td>
<td>Digital Pin 3 (PWM)</td>
<td>D3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>PD4 (PCINT20/XCK/T0)</td>
<td>Digital Pin 4</td>
<td>SD card Slave Select Line</td>
<td>D4</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>VCC</td>
<td>VCC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>GND</td>
<td>GND</td>
<td>GND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>PB6 (PCINT6/XTAL1/TOSC1)</td>
<td>Crystal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>PB7 (PCINT7/XTAL2/TOSC2)</td>
<td>Crystal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>PD5 (PCINT21/OC0B/T1)</td>
<td>Digital Pin 5 (PWM)</td>
<td>PWM Output for Motor 1</td>
<td>D5</td>
<td>5</td>
</tr>
<tr>
<td>12</td>
<td>PD6 (PCINT22/OC0A/AIN0)</td>
<td>Digital Pin 6 (PWM)</td>
<td>PWM Output for Motor 2</td>
<td>D6</td>
<td>6</td>
</tr>
<tr>
<td>13</td>
<td>PD7 (PCINT23/AIN1)</td>
<td>Digital Pin 7</td>
<td>Direction Output for Motor 2</td>
<td>D7</td>
<td>7</td>
</tr>
<tr>
<td>14</td>
<td>PB0 (PCINT0/CLKO/ICP1)</td>
<td>Digital Pin 8</td>
<td>Direction Output for Motor 1</td>
<td>D8</td>
<td>8</td>
</tr>
<tr>
<td>15</td>
<td>PB1 (OC1A/PCINT1)</td>
<td>Digital Pin 9</td>
<td>Grove Connector Output</td>
<td>D9</td>
<td>9</td>
</tr>
<tr>
<td>16</td>
<td>PB2 (SS/OC1B/PCINT2)</td>
<td>Digital Pin 10</td>
<td>Grove Connector Output</td>
<td>D10</td>
<td>10</td>
</tr>
<tr>
<td>17</td>
<td>PB3 (MOSI/OC2A/PCINT3)</td>
<td>Digital Pin 11</td>
<td>Data In for SD Card</td>
<td>D11</td>
<td>11</td>
</tr>
<tr>
<td>18</td>
<td>PB4 (MISO/PCINT4)</td>
<td>Digital Pin 12</td>
<td>Data Out for SD Card</td>
<td>D12</td>
<td>12</td>
</tr>
<tr>
<td>19</td>
<td>PB5 (SCK/PCINT5)</td>
<td>Digital Pin 13</td>
<td>Serial Clock Line for SD Card</td>
<td>D13</td>
<td>13</td>
</tr>
<tr>
<td>20</td>
<td>AVCC</td>
<td>VCC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>AREF</td>
<td>Analog Reference</td>
<td>AREF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>GND</td>
<td>GND</td>
<td>GND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>PC0 (ADC0/PCINT8)</td>
<td>Analog Input 0</td>
<td>Grove Connector Output</td>
<td>A0</td>
<td>A0</td>
</tr>
<tr>
<td>24</td>
<td>PC1 (ADC1/PCINT9)</td>
<td>Analog Input 1</td>
<td>Grove Connector Output</td>
<td>A1</td>
<td>A1</td>
</tr>
<tr>
<td>25</td>
<td>PC2 (ADC2/PCINT10)</td>
<td>Analog Input 2</td>
<td>MR Sensor Output</td>
<td>A2</td>
<td>A2</td>
</tr>
<tr>
<td>26</td>
<td>PC3 (ADC3/PCINT11)</td>
<td>Analog Input 3</td>
<td>FSR Output</td>
<td>A3</td>
<td>A3</td>
</tr>
<tr>
<td>27</td>
<td>PC4 (ADC4/SDA/PCINT12)</td>
<td>Analog Input 4</td>
<td></td>
<td>A4</td>
<td>A4</td>
</tr>
<tr>
<td>28</td>
<td>PC5 (ADC5/SCL/PCINT13)</td>
<td>Analog Input 5</td>
<td></td>
<td>A5</td>
<td>A5</td>
</tr>
</tbody>
</table>
Graphics for Hapkit (using Processing)
Processing for graphics

**Processing** is a programming language and a development environment, similar to Arduino.

With Processing, you can relatively easily create programs that run on your computer and communicate via the Serial line with your Hapkit Board.

For example, you can generate graphics that correspond to what your Hapkit is doing.
Steps to graph Hapkit position

1. Download the Processing development environment from [https://processing.org/download/](https://processing.org/download/), and install it.

2. Create a processing sketch that reads in 1-D data from the serial port and plots it to the screen (for example, as a dot with changing x position or as a time trace)

3. Print the variable you want to display in your Hapkit/Arduino sketch, e.g. `Serial.println(xh);`
Processing Demo

Arduino and Processing (Graph Example)
Assignment 5 due 5/9

Mid-quarter teaching evaluations

Suggested (optional) readings

1. Adding Graphics

2. Teleoperation Implementation
done in teams of two!

3. Teleoperation Analysis