→ We have seen simple control strategies to address tasks that require:

- **Motion Control** (e.g., resolved motion control using $(J^{-1})$ differential kinematics)
- **Static Force Control** (e.g., resisting a force at the end effector $(J^T)$)

→ In practice, many tasks that require physical interaction require a combination of both.

For example: **Drawing**

![Diagram of a robot arm with a pen attached to the end effector.](image)

The task in this case is described by motions/trajectory in some dimensions $(x, y)$, but is better described in terms of force in the $z$ dimension.

→ Adjust pen pressure
→ Adapt to small uncertainties or irregularities in the height of the drawing plane.

→ How would we control a task where we care both about force and velocity?
Unfortunately it is not easy or possible sometimes to control both force and position. (They can give conflicting feedback.)

Hybrid force/position control is a simpler alternative to do so, based on separating regions of space between velocity and force.

(Attributed to Matthew Mason 1981)

Consider the example of pulling a peg out of a hole.

How do we describe this task?
Let's define first a reference frame attached to the task (in this case the hole on the environment. Sometimes this is referred as C-frame (CONSTRAINT FRAME).

The idea is to define the task by a set of Kinematic and Static constraints:

<table>
<thead>
<tr>
<th>Kinematic Constraints (Motions)</th>
<th>Static Constraints (forces)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( V_x = 0 )</td>
<td>( f_2 = 0 )</td>
</tr>
<tr>
<td>( V_y = 0 )</td>
<td>( f_2 = 0 )</td>
</tr>
<tr>
<td>( W_x = 0 )</td>
<td></td>
</tr>
<tr>
<td>( W_y = 0 )</td>
<td></td>
</tr>
<tr>
<td><strong>Artificial Constraints</strong></td>
<td></td>
</tr>
<tr>
<td>( w_2 = 0 )</td>
<td>( f_x = 0 )</td>
</tr>
<tr>
<td>( w_2 = +V )</td>
<td>( z_2 = 0 )</td>
</tr>
<tr>
<td></td>
<td>( f_y = 0 )</td>
</tr>
<tr>
<td></td>
<td>( z_y = 0 )</td>
</tr>
</tbody>
</table>

Note: We cannot exert force in this direction if there is no friction.

We differentiate between Natural Constraints (those determined by the physics of contact) and Artificial Constraints (those we determine at will).

Note: This will assure we are in a region of the space where we can control the manipulator at will (no singularity).

In order to generalize this idea we define:

\[ V = \text{space of possible velocities (or forces)} \]

In the case of a spatial manipulator, this is \( IR^6 \) \((v_x, v_y, v_z, w_x, w_y, w_z)\).
Now $V_A$ = ADMISSIBLE = Subspace of $V$ of admissible velocities that conform to the natural constraints of the task

$V_C$: CONSTRAINT = $V_A^\perp$ (orthogonal complement of $V_A$)

$\forall v \in V \ s.t. \ v \cdot w = 0 \ \forall w \in V_A$

E.g.

\[
\begin{cases}
V_A = \text{plane } x, y \\
V_C = \text{axis } z.
\end{cases}
\]

we can write that $V = V_A \oplus V_C$

meaning that all elements of $V$ can be written as:

$\vec{v} = \vec{v}_A + \vec{v}_C$ with $\vec{v}_A \in V_A$ and $\vec{v}_C \in V_C$

$\rightarrow$ As a rule of thumb we will say that a task has:

CONSTRAINED DIRECTIONS ($V_C$) $\rightarrow$ No displacement allowed (zero velocity)

- We will use FORCE control mode in these directions and the task will be defined as a reference force trajectory.

UNCONSTRAINED DIRECTIONS ($V_A$) $\rightarrow$ No reaction force from the environment (zero force)

- We will use POSITION control mode in these directions and the task will be defined as a reference position/velocity trajectory.
**Simple Example:**

- **Move in a Slot:**

<table>
<thead>
<tr>
<th>Motion</th>
<th>Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>NATURAL: $\mathbf{v}_y = 0$, $\mathbf{f}_x = 0$</td>
<td></td>
</tr>
<tr>
<td>ARTIFICIAL: $\mathbf{v}_x = \mathbf{k}$, $\mathbf{f}_y = 0$</td>
<td></td>
</tr>
</tbody>
</table>

In this case $\mathbf{v}_A = \mathbf{\frac{1}{2}} \mathbf{v}_c = (a, 0)$

$\mathbf{v}_C = \mathbf{\frac{1}{2}} \mathbf{v}_c = (a, c)$

$\mathbf{v} = \mathbf{v}_A \oplus \mathbf{v}_C$ because any velocity can be written as $\mathbf{v} = \mathbf{v}_a + \mathbf{v}_c$ in $\mathbf{v}_A$ with $(\mathbf{v}_c = (0, c) = (0, 1))$

It will be the case that the same is true for forces:

$\mathbf{f} = \mathbf{f}_a + \mathbf{f}_c = \mathbf{f}_c$ in $\mathbf{v}_C$

0 because the natural constraint $\mathbf{f}_x = 0$

**Example:** SCREW DRIVER

<table>
<thead>
<tr>
<th>Motion</th>
<th>Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>NATURAL: $\mathbf{v}_x = 0$, $\mathbf{v}_c$, $\mathbf{f}_y = 0$, $\mathbf{v}_A$</td>
<td>IMPOSED PHYSICS</td>
</tr>
<tr>
<td>ARTIFICIAL: $\mathbf{w}_y = 0$, $\mathbf{w}_z = 0$, $\mathbf{f}_x = 0$, $\mathbf{v}_A$, $\mathbf{f}_z$ = $-\mathbf{N}$, $\mathbf{f}_z$ = $0$</td>
<td>FREE TO CHOOSE</td>
</tr>
</tbody>
</table>
Example: NUT DRIVER

\[ \begin{array}{|c|c|c|} \hline
\text{MOTION} & \text{FORCE} \\
\hline
\text{NATURAL} & \begin{align*}
V_x &= 0 \\
V_y &= 0 \\
V_z &= 0 \\
& \text{\( \eta \)} \\
\end{align*} & \begin{align*}
\tau_x &= 0 \\
\tau_y &= 0 \\
\tau_z &= \eta \\
& \text{\( F_a \)} \\
\end{align*} \\
\hline
\text{ARTIFICIAL} & \begin{align*}
W_x &= \alpha \\
W_y &= \gamma \\
W_z &= \beta \\
& \text{\( F_x \)} \\
\end{align*} & \begin{align*}
\tau_x &= 0 \\
\tau_y &= 0 \\
\tau_z &= \eta \\
& \text{\( F_a \)} \\
\end{align*} \\
\hline
\end{array} \]

**NOTE:** Which one is easier to use?

- The bolt/nut driver is more naturally constrained in motion, so for us it is easier. We are really good at freeing motion and use force instead to control.

- For robots, force control has traditionally been more difficult. It requires expensive sensors and very high frequency control loops. Actually there are very few industrial robots that do force or torque control. Instead, they are really good at position control.

→ In general, the axes of the task reference frame do not need to be aligned with the directions used to partition FORCE-MOTION constraints.

Fortunately, the orthogonality properties still hold, which is known as MASON’S PRINCIPLE.

**Assuming:**

- No friction
- Quasi-static interaction
NOTE: What is quasi-static?

- The dynamic components of the motion are negligible, which means that the instantaneous motion is determined by the current force, but can assume velocity is zero.

- Very common assumption in robotic manipulation:
  - I can do things faster or slower, the outcome is the same.
  - I can stop execution and restart and the motion won’t be affected.
  - If forces stop → STATIC equilibrium immediately.

If \( V_a \) is the set of admissible motions, and \( V_c \) is its orthogonal complement, then the structure of constraints, with

\[
\dot{p} = \dot{p}_c + \dot{p}_a \\
\dot{f} = \dot{f}_c + \dot{f}_a \\
V_c \quad V_a \\
V_c \quad V_a
\]

Then:

<table>
<thead>
<tr>
<th>MOTION</th>
<th>FORCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>NATURAL</td>
<td>( \dot{p}_c = 0 )</td>
</tr>
<tr>
<td>ARTIFICIAL</td>
<td>( \dot{p}_a ) arbitrary</td>
</tr>
</tbody>
</table>

Proof: The first column is clear, since it follows from the definition of \( V_a \).

For any virtual displacement of the system, the virtual work produced is:
\[ S.W = H^T \cdot \delta p \]

Endpoint force/moment infinitesimal acting on the object displacement

\[ S.W = (f_a + f_c) \cdot (\delta p_a + \delta p_c) = \]

\[ = f_a^T \delta p_a + f_a^T \delta p_c + f_c^T \delta p_a + f_c^T \delta p_c = \]

Because \( V_a + V_c \) by definition

\[ = f_a^T \delta p_a + f_c^T \delta p_c = \]

\[ 0 \]

by assumption and definition of \( V_c \)

There is no motion inside \( V_c \)

\[ = f_a^T \delta p_a = 0 \]

Because by assumption the process is quasi-static, and in that case, all virtual motions \( \delta p_a \) must produce zero virtual work.

This must be satisfied by virtue of the principle of virtual work \( \rightarrow \text{NATURAL CONSTRAINT} \)

This is true for all values of \( f_c \), which means that it becomes an \text{ARTIFICIAL CONSTRAINT} \#

How does a control architecture look like then?
+ Kinematics \( g \rightarrow p \)

**Position Sensor**

\[ \text{Derived (t)} + \text{Position (in } V_A) \rightarrow S \rightarrow \Delta p_A \rightarrow JI^{-1} \rightarrow \Delta q_a \rightarrow \text{PID} \]

\[ \text{PI D} \rightarrow \text{Robot} \rightarrow \text{Environment} \]

\[ \text{Desired (t)} + \text{Force (in } V_c) \rightarrow I - S \rightarrow \Delta f_c \rightarrow J^T \rightarrow \Delta z_c \rightarrow \text{PID} \]

\[ \text{FORCE SENSOR} \]

specified in the task

This structure avoids conflicts between natural constraints and self-imposed task constraints.

\[ \rightarrow \text{Pa is a projection matrix into the } V_A \text{ subspace to make sure the two components remain isolated} \]

\[ \text{e.g.: for the task of removing pin from hole:} \]

\[ Pa = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]

\[ Pc = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \]

\[ \rightarrow \text{Note also that this approach is not valid for dynamic tasks where we might want to control both position and force (juggling, throwing, running, ...)} \]