Hybrid Force/Motion Control for Robotics: A Comprehensive Guide

From Theory to Implementation

Table of Contents

1. Introduction to Hybrid Control

When a robot interacts with its environment, it faces a fundamental challenge: some directions allow free motion, while others are constrained by contact. Consider a simple task like pushing a box across a table—you can move freely along the table's surface, but you cannot penetrate through it. This duality between motion control and force control lies at the heart of **hybrid force/motion control**.

The Core Problem

Traditional robot control focuses either on **position control** (moving the end-effector to desired locations) or **force control** (applying specific forces). However, real-world tasks often require both simultaneously:

- Polishing a surface: follow a path (motion) while maintaining constant pressure (force)
- Assembly operations: insert a peg while controlling contact forces
- Surface finishing: track a contour while regulating normal force

The key insight of hybrid control is that **force and motion control should never act in the same direction**. Attempting to control both position and force along the same axis creates conflicting commands and unstable behavior.

Historical Context

The hybrid force/motion control framework was developed to handle robot-environment interaction when the environment is **infinitely stiff** (or nearly so). This contrasts with impedance control, which assumes compliant contacts. The hybrid approach explicitly separates task space into:

1. **Constrained directions**: Where the environment prevents motion and the robot must control forces
2. **Free directions**: Where the robot can move freely and should control motion

Key Advantages

Compared to earlier approaches using constrained dynamics, hybrid control offers:

- **Direct task specification**: Define constraints naturally using a task frame
- **Geometric filtering**: Handle non-ideal conditions (friction, compliance, uncertainties) by filtering measurements
- **Conflict avoidance**: Never attempt to control force and motion in the same direction
- **Practical robustness**: Filter out spurious measurement components that shouldn't exist according to the ideal model

2. Natural and Artificial Constraints

The hybrid control framework distinguishes between two types of constraints that completely characterize a task.

Natural Constraints

Natural constraints arise from the geometry and physics of the contact. They describe what is physically possible or impossible.

For a task involving contact with a rigid, frictionless environment, we can identify directions in **6D task space** (3 translational + 3 rotational):

Type	amp; **Count**	amp; **Description**
Motion constraints	amp; $6-k$	amp; End-effector cannot move (linear v or angular ω)
Force constraints	amp; k	amp; Environment cannot generate reaction forces/torques

Table 1: Natural constraint structure

The complementary nature is crucial: if motion is prevented in a direction, the environment **can** generate reaction forces there. Conversely, if the environment **cannot** generate forces in a direction, motion is **free** there.

Mathematical Expression:

For a 6D twist (generalized velocity):

$$V = \begin{bmatrix} v \\ \omega \end{bmatrix}$$

and a 6D wrench (generalized force):

$$F = \begin{bmatrix} f \\ \mu \end{bmatrix}$$

Natural constraints ensure that reaction forces do no work on feasible motions:

$$F^T V = 0$$

This orthogonality condition is fundamental to the hybrid control framework.

Artificial Constraints

Artificial constraints specify how we want to execute the task. They define the reference values for the control system.

Along the k free motion directions, we specify:

- Desired velocities: $v_{\text{des}}, \omega_{\text{des}}$

Along the $6-k$ constrained force directions, we specify:

- Desired forces/torques: $f_{\text{des}}, \mu_{\text{des}}$

Important: These are **our choices** as task designers. For example:

- To avoid internal stress, we might set unused force components to zero
- To maintain contact, we might command a normal force
- To execute motion, we specify desired speeds

The term "artificial" can be misleading—these are simply the task specifications we impose through control, complementary to the natural geometric constraints.

3. Task Frame Formalism

The **task frame** RF_t is the key to organizing hybrid control. It provides a coordinate system where the separation between motion and force directions is explicit.

Definition and Placement

The task frame is attached at the **contact point** or **end-effector**, with axes oriented such that:

- Each axis direction aligns with either pure motion or pure force control
- The frame may move and rotate as the task progresses
- Orientation is chosen to minimize coupling between directions

Coordinate Systems in Hybrid Control

A complete hybrid control implementation involves multiple frames:

Frame	amp; **Symbol**	amp; **Purpose**
Base frame	amp; RF_0	amp; Robot base reference, expresses $\dot{q}$ via Jacobian
Task frame	amp; RF_t	amp; Defines motion/force directions, errors computed here
Sensor frame	amp; RF_s	amp; Where force/torque measurements are taken
End-effector frame	amp; RF_e	amp; Attached to robot end-effector

Table 2: Reference frames in hybrid control

Critical Point: All quantities (velocities, forces, errors) must be expressed in the **same frame**—typically the task frame—before computing control errors. This requires rotation transformations between frames.

The 12 Quantities

In the task frame, we can define 12 quantities:

Linear components (along x_t, y_t, z_t):

- Velocities: v_x, v_y, v_z
- Forces: f_x, f_y, f_z

Angular components (around x_t, y_t, z_t):

- Angular velocities: $\omega_x, \omega_y, \omega_z$
- Torques: μ_x, μ_y, μ_z

The natural constraints will assign 6 of these to specific values (typically zero for motion in constrained directions). The artificial constraints specify desired values for the remaining 6.

4. Mathematical Foundations

Screws, Twists, and Wrenches

Hybrid control uses the elegant mathematical framework of **screw theory** to unify treatment of linear and angular quantities.

Twist (generalized velocity):

$$V = \begin{bmatrix} v \\ \omega \end{bmatrix} \in \mathbb{R}^6$$

where $v \in \mathbb{R}^3$ is linear velocity and $\omega \in \mathbb{R}^3$ is angular velocity.

Wrench (generalized force):

$$F = \begin{bmatrix} f \\ \mu \end{bmatrix} \in \mathbb{R}^6$$

where $f \in \mathbb{R}^3$ is force and $\mu \in \mathbb{R}^3$ is torque/moment.

Power (work rate) is the natural pairing:

$$P = F^T V = f^T v + \mu^T \omega$$

Selection Matrices: T and Y

The parametrization of feasible motions and feasible reactions uses two complementary matrices.

Matrix $T(s)$: Parametrizes feasible motions

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = T(s)\dot{s}$$

where:

- $T(s) \in \mathbb{R}^{6 \times k}$ has k columns
- $s \in \mathbb{R}^k$ are motion parameters
- $\dot{s}$ represents the k free motion speeds

Matrix $Y(s)$: Parametrizes feasible reactions

$$\begin{bmatrix} f \\ \mu \end{bmatrix} = Y(s)\lambda$$

where:

- $Y(s) \in \mathbb{R}^{6 \times (6-k)}$ has $6 - k$ columns
- $\lambda \in \mathbb{R}^{6-k}$ are force/torque parameters
- λ represents the reaction force intensities

Orthogonality Condition:

The fundamental relationship ensuring reaction forces do no work on feasible motions:

$$T^T(s)Y(s) = 0$$

This $k \times (6 - k)$ matrix is identically zero, expressing that the column spaces of T and Y are orthogonal.

Simple vs. Generalized Screw Directions

Simple case: When T and Y are formed by selecting columns from the 6×6 identity matrix.

Example: Motion only along x_t and rotation around y_t:

$$T = \begin{bmatrix} 1 & amp; 0 \\ 0 & amp; 0 \\ 0 & amp; 0 \\ 0 & amp; 0 \\ 0 & amp; 1 \\ 0 & amp; 0 \end{bmatrix}$$

Generalized case: Columns of T and Y combine multiple directions, representing coupled screw motions (simultaneous translation and rotation).

Example: Screw insertion with pitch p:

$$T = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ \frac{2\pi}{p} \end{bmatrix}$$

This single column captures that linear motion v_z and angular motion ω_z are coupled: $\omega_z = \frac{2\pi}{p} v_z$.

5. Practical Examples

Example 1: Sliding a Cube Along a Guide

Task: A gripper firmly holds a cube that slides along a linear guide.

Figure 1: Task frame for cube sliding: x_t along guide, y_t against wall, z_t against floor

Task frame placement:

- x_t: along the guide direction
- y_t: perpendicular to guide, toward side wall
- z_t: perpendicular to guide, downward toward floor

Natural constraints:

Motion constraints	amp; **Force constraints**
$v_y = 0$ (cannot move into wall) $v_z = 0$ (cannot move into floor) $\omega_x = 0$ (cannot tip) $\omega_z = 0$ (cannot twist)	amp; $f_x = 0$ (no friction) amp; $\mu_y = 0$ (free to slide) amp; amp;

Table 3: Natural constraints for cube sliding ($k = 2$, $6 - k = 4$)

Artificial constraints (our choices):

- $v_x = v_{x,\text{des}}$ — sliding speed (could be time-varying trajectory)
- $\omega_y = 0$ — don't rotate around guide (maintain orientation)
- $f_y = 0$ — don't push against side wall (avoid stress)
- — push down to maintain contact
- $\mu_x = 0$, $\mu_z = 0$ — don't apply unnecessary torques

Parametrization:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} 1 & amp; 0 \\ 0 & amp; 0 \\ 0 & amp; 0 \\ 0 & amp; 0 \\ 0 & amp; 1 \\ 0 & amp; 0 \end{bmatrix} \begin{bmatrix} v_x \\ \omega_y \end{bmatrix} = T \begin{bmatrix} v_x \\ \omega_y \end{bmatrix}$$

$$\begin{bmatrix} f \\ \mu \end{bmatrix} = \begin{bmatrix} 0 & amp; 0 & amp; 0 & amp; 0 \\ 1 & amp; 0 & amp; 0 & amp; 0 \\ 0 & amp; 1 & amp; 0 & amp; 0 \\ 0 & amp; 0 & amp; 1 & amp; 0 \\ 0 & amp; 0 & amp; 0 & amp; 0 \\ 0 & amp; 0 & amp; 0 & amp; 1 \end{bmatrix} \begin{bmatrix} f_y \\ f_z \\ \mu_x \\ \mu_z \end{bmatrix} = Y \begin{bmatrix} f_y \\ f_z \\ \mu_x \\ \mu_z \end{bmatrix}$$

Verification: $T^T Y = 0$ (the 2×4 zero matrix).

Note: T and Y are constant matrices for this task.

Example 2: Turning a Crank

Task: Rotate a crank with a free-spinning handle.

Figure 2: Task frame for crank turning: rotates with handle, y_t tangent to circular path

Task frame placement:

- Origin: at handle grip point
- y_t: tangent to circular path (rotation direction)
- x_t: radial direction (toward/away from crank center)
- z_t: along crank rotation axis
- Frame rotates by angle α as task progresses

Natural constraints:

Motion constraints	amp; **Force constraints**
$v_x = 0$ (cannot move radially)	amp; $f_y = 0$ (no friction in tangent direction)
$v_z = 0$ (cannot move axially)	amp; $\mu_z = 0$ (handle spins freely)
$\omega_x = 0$ (cannot tilt handle)	amp;
$\omega_y = 0$ (cannot bend handle)	amp;

Table 4: Natural constraints for crank turning ($k = 2$, $6 - k = 4$)

Artificial constraints:

- $v_y = v_{y,\text{des}}$ — tangential speed (controls rotation rate)
- $\omega_z = 0$ (or $\omega_{z,\text{des}} \neq 0$) — handle spin behavior
- $f_x = 0$, $f_z = 0$ — don't pull/push radially or axially (avoid stress)
- $\mu_x = 0$, $\mu_y = 0$ — don't apply bending/tilting torques

Parametrization (expressed in base frame RF_0):

$$\begin{bmatrix} {}^0v \\ {}^0\omega \end{bmatrix} = \begin{bmatrix} R_z(\alpha) & amp; 0 \\ 0 & amp; R_z(\alpha) \end{bmatrix} \begin{bmatrix} 0 & amp; 0 \\ 1 & amp; 0 \\ 0 & amp; 0 \\ 0 & amp; 0 \\ 0 & amp; 0 \\ 0 & amp; 1 \end{bmatrix} \begin{bmatrix} v_y \\ \omega_z \end{bmatrix} = T(\alpha) \begin{bmatrix} v_y \\ \omega_z \end{bmatrix}$$

where $R_z(\alpha)$ is the rotation matrix about the z-axis by angle α.

Similarly for forces (with same rotation):

$$\begin{bmatrix} {}^0 f \\ {}^0 \mu \end{bmatrix} = Y(\alpha) \begin{bmatrix} f_x \\ f_z \\ \mu_x \\ \mu_y \end{bmatrix}$$

Key point: $T(\alpha)$ and $Y(\alpha)$ are **time-varying** as the task progresses, but the orthogonality $T^T(\alpha)Y(\alpha) = 0$ always holds (rotation matrices cancel).

Example 3: Inserting a Screw into a Bolt

Task: Insert a screw with pitch p into a threaded hole.

Figure 3: Task frame for screw insertion: z_t along insertion axis

Task frame placement:

- z_t: along insertion axis (downward)
- x_t, y_t: perpendicular to insertion

Natural constraints (partial listing):

- $v_x = 0$, $v_y = 0$ — cannot move perpendicular to threads
- $\omega_x = 0, \omega_y = 0$ — cannot tilt screw
- But: v_z and ω_z are **coupled** by pitch!

Key insight: This is not 2 independent DOF, but **1 DOF** due to the screw geometry:

$$\omega_z = \frac{2\pi}{p} v_z$$

If pitch $p = 0.5$ mm and we advance at $v_z = 0.5$ mm/s, then $\omega_z = 2\pi$ rad/s (one revolution per second).

Artificial constraints (redundant as initially stated):

Initially, one might write:

- $v_z = v_{z,\text{des}}$
- $\omega_z = \omega_{z,\text{des}}$

But these **cannot be independent**! The pitch constraint couples them.

Correct parametrization:

Only **one** free parameter, say v_z:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ \frac{2\pi}{p} \end{bmatrix} v_z = T\, v_z$$

This single column represents a **screw motion** (not a simple direction).

For forces, we need 5 independent directions orthogonal to T:

$$Y = \begin{bmatrix} 1 & amp;0 & amp;0 & amp;0 & amp;0 \\ 0 & amp;1 & amp;0 & amp;0 & amp;0 \\ 0 & amp;0 & amp;0 & amp;0 & amp;-\frac{2\pi}{p} \\ 0 & amp;0 & amp;1 & amp;0 & amp;0 \\ 0 & amp;0 & amp;0 & amp;1 & amp;0 \\ 0 & amp;0 & amp;0 & amp;0 & amp;1 \end{bmatrix}$$

Parameters: $\lambda = [f_x, f_y, \mu_x, \mu_y, \mu_z]^T$.

Notice the last column couples f_z and μ_z:

$$f_z = -\frac{2\pi}{p}\mu_z$$

This ensures $T^T Y = 0$: forces along the screw direction do no work on screw motion.

Artificial constraints (corrected):

- $v_z = v_{z,\text{des}}$ — insertion speed (the single motion DOF)
- $f_x = 0$, $f_y = 0$ — don't push sideways
- $\mu_x = 0$, $\mu_y = 0$ — don't tilt
- μ_z (or equivalently f_z) — can specify either, not both independently

Typically, we'd specify the downward force $f_z = f_{z,\text{des}}$, and the required torque $\mu_z = -\frac{p}{2\pi} f_{z,\text{des}}$ follows automatically from the screw geometry.

6. Control Law Design

Feedback Linearization in Task Space

The control objective is to impose desired behaviors on:

- Motion parameters: $s \to s_d(t)$
- Force parameters: $\lambda \to \lambda_d(t)$

Robot dynamics with contact forces:

$$M(q)\ddot{q} + S(q,\dot{q})\dot{q} + g(q) = u + J^T(q)\begin{bmatrix} f \\ \mu \end{bmatrix}$$

where:

- $M(q)$: inertia matrix
- $S(q,\dot{q})\dot{q}$: Coriolis and centrifugal terms
- $g(q)$: gravity
- u: joint torques
- $J(q)$: geometric Jacobian
- $[f^T, \mu^T]^T$: wrench at end-effector

Task-space relationships:

From robot kinematics:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = J(q)\dot{q}$$

From task parametrization:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = T(s)\dot{s}$$

$$\begin{bmatrix} f \\ \mu \end{bmatrix} = Y(s)\lambda$$

Differentiating the motion equation:

$$J\ddot{q} + \dot{J}\dot{q} = T\ddot{s} + \dot{T}\dot{s}$$

Solving for $\ddot{q}$ (assuming J non-singular):

$$\ddot{q} = J^{-1}(T\ddot{s} + \dot{T}\dot{s} - \dot{J}\dot{q})$$

Substituting into robot dynamics:

$$MJ^{-1}(T\ddot{s} + \dot{T}\dot{s} - \dot{J}\dot{q}) + S\dot{q} + g = u + J^T Y\lambda$$

Choosing the control law:

$$u = \left[MJ^{-1}T \quad amp; \vdots \quad amp; -J^T Y\right] \begin{bmatrix} a_s \\ a_\lambda \end{bmatrix} + MJ^{-1}(\dot{T}\dot{s} - \dot{J}\dot{q}) + S\dot{q} + g$$

This yields the **linearized and decoupled** closed-loop dynamics:

$$\begin{bmatrix} \ddot{s} \\ \lambda \end{bmatrix} = \begin{bmatrix} a_s \\ a_\lambda \end{bmatrix}$$

Key observations:

- Motion parameters s have **relative degree 2** (acceleration input)
- Force parameters λ have **relative degree 0** (direct input)
- The matrix $[MJ^{-1}T \vdots - J^T Y]$ is non-singular if JTY has full column rank

Stabilization Strategy

With the linearized system, we can design simple linear controllers for each decoupled channel.

For motion (k channels):

$$a_s = \ddot{s}_d + K_D(\dot{s}_d - \dot{s}) + K_P(s_d - s)$$

This gives closed-loop error dynamics:

$$\ddot{e}_s + K_D\dot{e}_s + K_P e_s = 0$$

where $e_s = s_d - s$.

Choose (diagonal positive definite) for exponential stability. This is a **PD controller with feedforward**.

For force ($6 - k$ channels):

$$a_\lambda = \lambda_d + K_I \int (\lambda_d - \lambda)\, dt$$

This gives:

$$\dot{e}_\lambda + K_I \int e_\lambda \, dt = 0$$

where $e_\lambda = \lambda_d - \lambda$.

Choose $K_I \geq 0$ (diagonal). The integral term provides robustness against constant force disturbances.

Note: Since λ has relative degree 0, simple proportional control $a_\lambda = \lambda_d$ would suffice for tracking, but integral action improves disturbance rejection.

Alternative: Force control via impedance model

When contact stiffness is finite (not infinite), we can use an impedance model for force-controlled directions.

For a single direction with position x, contact point x_c, contact stiffness k_c, and desired force f_d:

$$m_i\ddot{x} + d_i\dot{x} + k_c(x - x_c) = f_d$$

The measured force is $f_m = k_c(x - x_c)$.

After feedback linearization ($\ddot{x} = a_x$):

$$a_x = \frac{1}{m_i}(f_d - f_m) - \frac{d_i}{m_i}\dot{x}$$

This is a **P controller** on force with velocity damping. Parameters and are design choices (not physical).

Benefits:

- Works during approach phase ($f_m = 0$): gives steady-state approach velocity $\dot{x}_{ss} = f_d/d_i$
- After contact: regulates force with impedance-like behavior

7. Geometric Filtering

A critical feature of hybrid control is **geometric filtering** of measurements. This handles non-ideal conditions that violate the ideal model assumptions.

The Measurement Problem

In the ideal model:

- Motion should occur **only** in free directions (no velocity in constrained directions)
- Forces should appear **only** in constrained directions (no force in free directions)

In reality, measurements show:

- Small velocities in constrained directions (due to compliance)
- Small forces in free directions (due to friction)

Problem: If we naively use these measurements to compute errors, we'd try to control force and motion in the same direction simultaneously!

Solution: Filter Measurements

Principle: Only use measurement components that **should** exist according to the ideal model. Treat others as disturbances.

For motion parameters s and $\dot{s}$:

Obtain from joint measurements using:

$$\dot{s} = T^{\dagger}(s)J(q)\dot{q}$$

where $T^{\dagger} = (T^T T)^{-1}T^T$ is the pseudoinverse.

This **projects** the measured velocity onto the feasible motion subspace, automatically filtering out components in constrained directions.

For force parameters λ:

Obtain from force sensor using:

$$\lambda = Y^{\dagger}(s)\begin{bmatrix} f \\ \mu \end{bmatrix}$$

This **projects** the measured wrench onto the feasible reaction subspace, filtering out components in free directions (e.g., friction forces).

Example: Planar Contact

Consider a 2D planar robot in pointwise contact with a surface.

Figure 4: Multiple frames: sensor measures in RF_s, velocity computed in RF_0, but control in task frame RF_t

Frame relationships:

- Sensor measures: ${}^sf = [f_x^s, f_y^s]^T$ in sensor frame
- Velocity computed: ${}^0v = [v_x^0, v_y^0]^T$ in base frame
- Task frame: x_t tangent to surface, y_t normal

Rotations required:

$${}^tf = R_{t\leftarrow s}\,{}^sf$$

$${}^tv = R_{t\leftarrow 0}\,{}^0v$$

where $R_{t\leftarrow s}$ and $R_{t\leftarrow 0}$ are 2D rotation matrices aligning frames.

Filtering in task frame:

Even if velocity is purely tangential (${}^tv = [v_x^t, 0]^T$ expected), compliance might give small $v_y^t \neq 0$.

The projection $\dot{s} = T^\dagger\,{}^tv$ automatically zeros this component:

$$T = \begin{bmatrix}1\\0\end{bmatrix}, \quad T^\dagger = [1 \quad amp; 0]$$

$$\dot{s} = [1 \quad amp; 0]\begin{bmatrix}v_x^t\\v_y^t\end{bmatrix} = v_x^t$$

Similarly for force: even if ideal model predicts ${}^tf = [0, f_y^t]^T$, friction gives $f_x^t \neq 0$.

The projection $\lambda = Y^\dagger\,{}^tf$ filters this:

$$Y = \begin{bmatrix}0\\1\end{bmatrix}, \quad Y^\dagger = [0 \quad amp; 1]$$

$$\lambda = [0 \quad amp; 1]\begin{bmatrix}f_x^t\\f_y^t\end{bmatrix} = f_y^t$$

Result: The controller sees only the components it should control, treating deviations as disturbances to be rejected by feedback.

8. Handling Non-Ideal Conditions

The geometric filtering approach addresses three main non-idealities:

1. Friction at the Contact

Effect: Friction generates forces opposing motion in "free" directions.

Example: Sliding a block—friction force f_{friction} opposes tangential motion.

Solution:

- The force projection $\lambda = Y^\dagger[f, \mu]^T$ ignores tangential force components
- Motion controller sees no conflicting force signal
- Friction acts as a disturbance to velocity tracking, rejected by feedback gains

Note: For Coulomb friction, tangential force depends on normal force: $|f_t| \leq \mu_f |f_n|$. This coupling can be handled but adds complexity.

2. Compliance in Structure/Contact

Effect: The environment or robot isn't perfectly rigid—small displacements occur in "constrained" directions.

Example: Pushing on a slightly compliant wall—the end-effector moves a bit.

Solution:

- The velocity projection $\dot{s} = T^\dagger J \dot{q}$ ignores velocity components in constrained directions
- Force controller regulates the commanded force despite small position drift
- Compliance acts as a disturbance, compensated by integral force control

Additional consideration: For stability with finite contact stiffness, the impedance-based force control (Section 6.2) explicitly models compliance:

$$m_i \ddot{x} + d_i \dot{x} + k_c(x - x_c) = f_d$$

This provides proper dynamics matching for the force loop.

3. Uncertainty in Surface Geometry

Effect: We don't know the exact orientation of the contact surface.

Example: Approaching an unknown surface—where is the normal direction?

Consequence: Task frame orientation is uncertain.

Solution approaches:

- **Vision-based estimation**: Use cameras to measure surface orientation before contact
- **Force-based estimation**: Normal direction aligns with measured force (assumes no friction)
- **Velocity-based estimation**: Tangent direction aligns with motion (assumes rigid contact)
- **Sensor fusion**: Combine force and position measurements using Recursive Least Squares (RLS)

Sensor fusion approach:

Given noisy measurements affected by both friction and compliance, estimate surface parameters θ (e.g., surface tangent angle) by:

$$\hat{\theta}(t) = \arg\min_{\theta} \sum_{i=0}^{t} w(t-i) \| r_i(\theta) \|^2$$

where $r_i(\theta)$ is the residual between measurements and model predictions, and $w(t-i)$ is a forgetting factor.

RLS update equations: Efficiently compute online updates as new measurements arrive.

9. Implementation Considerations

Control Architecture

The hybrid control system has a layered structure:

Figure 5: Block diagram showing inner (motion) and outer (force) loops with task-space feedback linearization

Components:

1. **Measurement filtering**:
   - Rotate force measurements to task frame: ${}^t f = R_{t \leftarrow s} \, {}^s f$
   - Rotate velocities to task frame: ${}^t v = R_{t \leftarrow 0} J(q) \dot{q}$
   - Extract parameters: $\dot{s} = T^\dagger(s) \, {}^t v$, $\lambda = Y^\dagger(s) \, {}^t F$
2. **Error computation** (in task frame):
   - Motion errors: $e_s = s_d - s$, $\dot{e}_s = \dot{s}_d - \dot{s}$

  - Force errors: $e_\lambda = \lambda_d - \lambda$

3. **Stabilizing controllers**:
   - $a_s = \ddot{s}_d + K_D \dot{e}_s + K_P e_s$ (k motion channels)
   - $a_\lambda = \lambda_d + K_I \int e_\lambda \, dt$ ($6 - k$ force channels)
4. **Feedback linearization**:
   - Compute u from a_s, a_λ using linearizing law
   - Requires: $M(q)$, $J(q)$, $\dot{J}(q)$, $S(q, \dot{q})$, $g(q)$, $T(s)$, $\dot{T}(s)$, $Y(s)$
5. **Robot execution**:
   - Send u to joint torque controllers
   - Measure q, $\dot{q}$ from encoders
   - Measure wrench from force/torque sensor

Selection Matrix Options

For simple tasks, T and Y are **selection matrices**: columns from I_6.

Example: Cube sliding (Example 1)

$$\Sigma = \begin{bmatrix} 1 & amp;0 \\ 0 & amp;0 \\ 0 & amp;0 \\ 0 & amp;0 \\ 0 & amp;1 \\ 0 & amp;0 \end{bmatrix}, \quad I_6 - \Sigma = \begin{bmatrix} 0 & amp;0 & amp;0 & amp;0 \\ 1 & amp;0 & amp;0 & amp;0 \\ 0 & amp;1 & amp;0 & amp;0 \\ 0 & amp;0 & amp;1 & amp;0 \\ 0 & amp;0 & amp;0 & amp;0 \\ 0 & amp;0 & amp;0 & amp;1 \end{bmatrix}$$

Alternative notation: Some literature uses binary selection matrices Σ and $I - \Sigma$ instead of T and Y.

Advantage: Simpler to visualize and implement when motion/force are uncoupled.

Limitation: Cannot represent generalized screw motions (e.g., screw insertion).

Practical Gain Tuning

Motion control gains (K_P, K_D):

- Start with critically damped: $K_D = 2\sqrt{K_P}$
- Increase K_P for stiffer position tracking (faster response)
- Reduce if oscillations occur (model uncertainties, unmodeled dynamics)
- Typical values: $K_P \sim 100$–1000 (units depend on scaling)

Force control gains (K_I):

- Start with low gain for stability
- Increase until steady-state force error is acceptable
- Too high → oscillations or instability with contact dynamics
- Typical values: $K_I \sim 1$–10 (slower than position loop)

General principle: Force loop should be **slower** than motion loop due to:

- Contact dynamics (compliance, sensor dynamics)
- Harder to measure forces accurately than positions
- Stability margins tighter for force control

Cascaded vs. Unified Implementations

Two architectural options:

1. Unified (single-loop): Feedback linearization at joint level, both motion and force controlled directly.

2. Cascaded (inner-outer):

- Inner loop: Fast position/velocity control at joint or Cartesian level
- Outer loop: Force control generates position/velocity references for inner loop

Trade-offs:

Aspect	amp; **Unified**	amp; **Cascaded**
Performance	amp; Higher (direct control)	amp; Lower (indirect, lag)
Stability	amp; Requires accurate model	amp; More robust
Complexity	amp; Higher (full dynamics)	amp; Lower (uses existing controllers)
Industry adoption	amp; Research systems	amp; Industrial robots

Table 5: Comparison of control architectures

Cascaded implementation: Force error generates a position/velocity reference for an existing motion controller. Simpler but introduces lag.

10. Experimental Results

The hybrid force/motion control framework has been validated extensively in laboratory and industrial settings.

Historic Experiments (1991-1992)

Platform: MIMO-CRF robot at University of Rome "La Sapienza"

Key demonstrations:

1. **Contour following with force regulation**:
    - Robot traces unknown curved surfaces
    - Maintains constant normal force (20 N)
    - Simultaneously estimates surface geometry
2. **Surface profile reconstruction**:
    - Film reel contour (radius 17 cm)
    - RLS estimation fusing force and position data
    - Continuous online update of task frame orientation
    - Tangent angle estimated within 7-8° accuracy
3. **Hybrid control with online estimation**:
    - Contour tracking and estimation performed simultaneously
    - Force peaks detected grooves in surface geometry
    - Demonstrated robustness to geometric uncertainties

Industrial Application: Robotic Deburring

Task: Remove excess Polyvinyl Butyral (PVB) from car windshield edges.

Challenges:

- Fabrication tolerances in windshield shape
- Variable material thickness
- Sharp edges requiring controlled force

Solution:

- Pre-programmed path (nominal windshield profile)
- Compliant deburring tool accommodates geometric variations
- Pneumatic actuator controls normal force independently
- Two cutting blades: actuated and spring-loaded
- Load cell measures 1D normal force
- On-board controller regulates force via pneumatic valve

Results:

- Consistent material removal despite geometric variations
- Independent force control prevents tool damage
- Hybrid control separates path following (motion) from pressing force (force)

Comparison: Four Contact Control Strategies

A comparative video demonstration (COMAU Smart robot, University of Naples, 1994) showed:

1. **Compliance control**: Active Cartesian stiffness without force sensor
    - Allows contact by softening impedance
    - No explicit force regulation
2. **Impedance control**: With force/torque sensor feedback
    - Explicit force measurement
    - Target impedance behavior
3. **Force control**: External loop providing references to internal position loop
    - Cascaded architecture
    - Slower but easier to implement
4. **Hybrid force/position control**: Full implementation as described
    - Simultaneous, independent force and motion control
    - Highest performance for constrained tasks

Observation: Each approach has its place—hybrid control excels when task naturally splits into motion and force directions.

Performance Metrics

Typical achieved performance:

Metric	amp; **Typical Value**
Force regulation accuracy	amp; ± 0.5–2 N
Position tracking accuracy	amp; ± 0.1–1 mm
Surface angle estimation error	amp; 5–10°
Force bandwidth	amp; 1–10 Hz
Motion bandwidth	amp; 10–50 Hz

Table 6: Experimental performance metrics

Factors affecting performance:

- Contact stiffness (stiffer = harder force control)
- Force sensor noise and bandwidth
- Model accuracy (dynamics, kinematics, friction)
- Sampling rate (100–1000 Hz typical)
- Actuator capabilities (torque limits, dynamics)

11. Conclusions

Summary of Key Concepts

Hybrid force/motion control provides a principled framework for robot-environment interaction when the environment imposes geometric constraints. The key insights are:

1. **Complementary spaces**: Task space naturally divides into motion-controlled directions (k DOF) and force-controlled directions ($6 - k$ DOF)
2. **Task frame formalism**: A well-chosen task frame makes this decomposition explicit and geometrically intuitive
3. **Orthogonality principle**: Reaction forces do no work on feasible motions ($T^T Y = 0$), ensuring consistent parametrization
4. **Geometric filtering**: Projecting measurements onto feasible subspaces prevents conflicting control commands and handles non-ideal conditions
5. **Feedback linearization**: Decouples motion and force channels, enabling independent controller design for each
6. **Robustness**: The framework gracefully handles friction, compliance, and geometric uncertainties through measurement filtering

Advantages Over Alternatives

Compared to pure position control:

- Explicit force regulation prevents excessive contact forces
- Accommodates geometric uncertainties and surface variations
- Enables force-sensitive operations (assembly, polishing, etc.)

Compared to pure impedance control:

- Direct force specification (not just impedance parameters)
- Clearer task specification for constrained motions
- Better suited for rigid contacts and structured environments

Compared to constrained dynamics approaches:

- More intuitive task frame formalism
- Explicit handling of non-idealities (friction, compliance, errors)
- Geometric filtering prevents control conflicts

Limitations and Extensions

Assumptions:

- Robot has sufficient DOF ($n \geq m$, typically $n = m = 6$)
- Jacobian non-singular (no kinematic singularities)
- Accurate dynamic model available
- Force/torque sensor available

Extensions:

- **Redundant robots** (): Use null-space control for additional objectives
- **Underactuated systems**: Modified control laws for reduced actuation
- **Adaptive control**: Online parameter estimation for uncertain dynamics
- **Learning-based**: Use data-driven methods to learn task frame orientations or model parameters
- **Multi-contact**: Extension to multiple simultaneous contacts (e.g., bimanual manipulation)

Practical Recommendations

For successful implementation:

1. **Start simple**: Begin with 2D or single-direction examples (1D force, 1D motion)
2. **Calibrate carefully**: Accurate force sensor calibration and frame transformations are critical
3. **Tune conservatively**: Start with low gains, increase gradually while monitoring stability
4. **Verify frame alignment**: Ensure all measurements are properly rotated to task frame before computing errors
5. **Monitor in real-time**: Plot forces, velocities, errors during execution to diagnose issues
6. **Use simulation**: Validate control laws in simulation before hardware experiments
7. **Plan for safety**: Implement force limits, collision detection, emergency stops

Current Research Directions

Ongoing research extends hybrid control to:

- **Learning from demonstration**: Automatically infer task frames and constraint structure from examples
- **Vision integration**: Real-time surface estimation using cameras and point clouds
- **Deformable objects**: Handling non-rigid environments (e.g., soft materials, fabrics)
- **Human-robot collaboration**: Shared control where human and robot coordinate force and motion
- **Extreme environments**: Space, underwater, or high-speed applications with unique constraints

Final Thoughts

Hybrid force/motion control represents a mature and powerful approach to robot contact tasks. Its theoretical elegance—the clean separation of motion and force, the orthogonality condition, the geometric filtering—translates into practical robustness and intuitive task specification.

For applications ranging from assembly and machining to medical robotics and teleoperation, understanding when and how to decompose a task into motion-controlled and force-controlled directions is an essential skill for robotics engineers.

The framework's 40+ years of development and industrial deployment demonstrate both its fundamental correctness and its practical utility. As robots increasingly work in unstructured environments and close contact with humans, the principles of hybrid control—safely managing forces while accomplishing motion objectives—will only grow in importance.

References

[1] De Luca, A. (2020). Robotics 2 - Hybrid Force/Motion Control. Lecture notes, Sapienza University of Rome.

[^2] Mason, M. T. (1981). Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(6), 418-432.

[^3] Raibert, M. H., & Craig, J. J. (1981). Hybrid position/force control of manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 103(2), 126-133.

[^4] Khatib, O. (1987). A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1), 43-53.

[^5] Eppinger, S. D., & Seering, W. P. (1987). Understanding bandwidth limitations in robot force control. *Proceedings of the IEEE International Conference on Robotics and Automation*, 904-909.

[^6] De Luca, A., & Manes, C. (1994). Hybrid force/position control for robots in contact with dynamic environments. *Intelligent Automation and Soft Computing*, 1(1), 1-20.

[^7] Siciliano, B., & Villani, L. (1999). *Robot Force Control*. Kluwer Academic Publishers.

[^8] Sciavicco, L., & Siciliano, B. (2000). *Modelling and Control of Robot Manipulators* (2nd ed.). Springer-Verlag.

[^9] Murray, R. M., Li, Z., & Sastry, S. S. (1994). *A Mathematical Introduction to Robotic Manipulation*. CRC Press.

[^10] Spong, M. W., Hutchinson, S., & Vidyasagar, M. (2006). *Robot Modeling and Control*. John Wiley & Sons.

⁂

1. 16_HybridControl.pdf