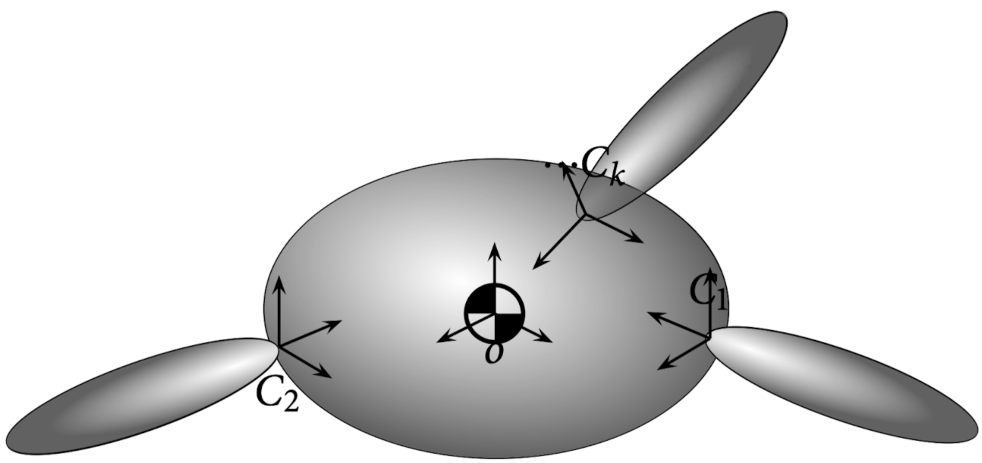
Lecture 19: Hand Dynamics and Control

Scribes: Michael Equi, Zach Tam

19.1 Grasp Statics

We can model a grasp with the mathematical group (G, FC). We define G as the matrix that defines how the frictional force FC are applied when the point contact makes a connection with the surface. Each column of G is represented in the contact frame as a 6-dimensional wrench where the z-axis points directly out from the contact point into the object.



In order to get the force imparted on the object by the point contact we transform the wrench into the world frame using the adjoint. An example of this on a frictionless point contact is shown below. Frictionless implies that the contact point only has a single wrench applied along the z-axis.

$$F_o = \begin{bmatrix} R_{ci} & 0 \\ \hat{p_{ci}} R_{ci} & R_{ci} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} x_i = \begin{bmatrix} n_{ci} \\ p_{ci} \times n_{ci} \end{bmatrix}, x_i \geq 0 \tag{19.1}$$

Since frictionless contacts are quite limiting and are required to be in large numbers as to provide adequate force closure we can extend the G matrix of each contact point and have x_i be a vector describing the friction applied in each of the frictioned axes. The following represents a model for a soft finger that provides friction in the x and y axis of the contact frame as well as resists rotational motion around the z axis of the contact frame.

$$F_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_i, x_i \in FC_i \tag{19.2}$$

In order to solve for a functional grasping position we can now need for find a vector x_i such that the total forces applied to the object is the desired F_o and each element of x_i corresponding to z-axis of the contact frame is positive. This leads to the definition of force closure.

A grasp (G, Fc) is **Force Closure** if $\forall F_o \in \mathbb{R}^p, \exists x \in FC$, s.t. $Gx = F_o$. This leaves us with the following three problems we need to solve in order to execute a grasp. (1) Determine if (G, FC) is force closure, (2) given F_o find $x \in FC$ s.t. $Gx = F_o$, and (3) satisfy (2) such that we additionally optimize for criterion $\Phi(x)$.

x is considered an **Internal Force** x_N if $x_N \in FC$ and $G_{x_N} = 0$. Using this definition for x_N we can state the following **property 3**; (G, FC) is force closure iff $G(FC) = \mathbb{R}^p$ and $\exists x_N \in ker(G)$ s.t. $x_N \in int(FC)$. Here $int(FC)$ denotes the interior region of the space defined by FC.

Property 4 describes equivalent statements about G that may be useful computationally for understanding force closure and performing the necessary optimization in order to solve for a xi that solves for the desired F_o. Let $G = G_1, G_2, ..., G_k$, then the following are equivalent:

1. (G, FC) are force closure
2. The columns of G span $\mathbb{R}^p$
3. The convex hull of G_i contains neighborhood of the origin
4. There does not exist a vector $v \in \mathbb{R}^p, v \neq 0$ s.t. $\forall i = 1, ..., k \ v \cdot G_i \geq 0$

19.2 Kinematics of Contact

19.2.1 Motivation

Often in robotics, it is necessary to deal with **deformable objects** such as towels (in folding) and organs (in surgery).

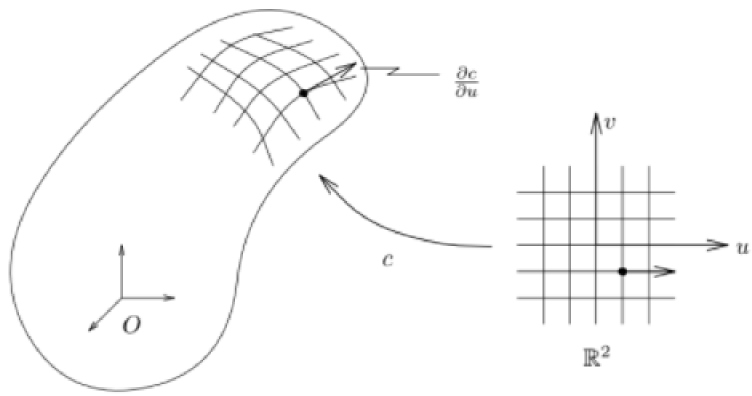
Force closure is the simplest grasp, but it is limited as it serves only to immobilize the object. However, multifingered hands are designed to be more versatile than this, and should be capable of **in-hand manipulation**. For example, in surgery, fine motions are required for blunt dissection (separating folds of skin), separating the bile duct from the liver duct when removing the gallbladder, and cauterizing/cutting. Even though the surgeon remains in control, the robot would need to manipulate the objects. As a result, we need to be able to model how moving the fingers causes the grasped object to move relative to the fingers.

To some extent, manipulability is the dual of grasping: there is an inherent tradeoff between manipulability and tightness of any grasp.

19.2.2 Surface Model

In order to model the surface contacts between fingers and grasped objects, we first need to be able to model surfaces.

To this end, we first define fixed planar Cartesian coordinates u and v in $\mathbb{R}^2$ as shown in the figure. We then map a subset U of the plane onto the surface we wish to model by using the map c:



$$\begin{aligned} &c : U \subset \mathbb{R}^2 \to \mathbb{R}^3, c(U) \subset S \\ &c_u = \frac{\partial c}{\partial u} \in \mathbb{R}^3 \\ &c_v = \frac{\partial c}{\partial v} \in \mathbb{R}^3 \end{aligned} \tag{19.3}$$

$c(u, v)$ covers the whole surface using all the points in U. Note that there are singularities in the partial derivatives at corners and edges.

For convenience, we define the first fundamental form I_p as:

$$I_p = \begin{bmatrix} {c_u}^T c_u & {c_u}^T c_v \\ {c_v}^T c_u & {c_v}^T c_v \end{bmatrix} \tag{19.4}$$

Under the assumption of **Orthogonal Coordinates Chart**, ${c_u}^T c_v = 0$ and thus:

$$I_p = \begin{bmatrix} ||c_u||^2 & 0 \\ 0 & ||c_v||^2 \end{bmatrix} \tag{19.5}$$

We define the metric tensor M_p to be the matrix square root of I_p under this assumption:

$$M_p = \begin{bmatrix} ||c_u|| & 0 \\ 0 & ||c_v|| \end{bmatrix} \tag{19.6}$$

The **Gauss Map** defines a normal to the surface at each point. s^2 is the surface of the unit sphere (defines a direction in 3D space).

$$N : S \to s^2 N(u,v) = \frac{c_u c_v}{||c_u c_v||} := n \tag{19.7}$$

Using the normal defined by the Gauss map, we can define the second fundamental form II_p as follows:

$$II_p = \begin{bmatrix} {c_u}^T n_u & {c_u}^T n_v \\ {c_v}^T n_u & {c_v}^T n_v \end{bmatrix}, n_u = \frac{\partial n}{\partial u}, n_v = \frac{\partial n}{\partial v} \tag{19.8}$$

We then define the curvature tensor K_p:

$$K_p = {M_p}^{-T} II_p {M_p}^{-1} = \begin{bmatrix} \frac{{c_u}^T n_u}{||c_u||^2} & \frac{{c_u}^T n_v}{||c_u||||c_v||} \\ \frac{{c_v}^T n_u}{||c_u||||c_v||} & \frac{{c_v}^T n_v}{||c_v||^2} \end{bmatrix} \tag{19.9}$$

The surface coordinates compose the **Gauss Frame**:

$$\begin{bmatrix} x & y & z \end{bmatrix} = \begin{bmatrix} \frac{c_u}{||c_u||} & \frac{c_v}{||c_v||} & n \end{bmatrix} \tag{19.10}$$

Thus, we can reparameterize K_p as:

$$K_p = \begin{bmatrix} x^T \\ y^T \end{bmatrix} \begin{bmatrix} \frac{n_u}{||c_u||} & \frac{n_v}{||c_v||} \end{bmatrix} \tag{19.11}$$

We can also define the torsion form T_p as:

$$T_p = y^T \begin{bmatrix} \frac{x_u}{||c_u||} & \frac{x_v}{||c_v||} \end{bmatrix} \tag{19.12}$$

The reason for all of this definition is to define (M_p, K_p, T_p) which is a geometric parameter for the surface at each point.

19.2.3 Example: Geometric parameters of a sphere in $\mathbb{R}^3$

Let u be the angle of inclination, traditionally $\frac{\pi}{2} - \phi$ in spherical coordinates, and let v be the angle of azimuth, traditionally θ. The radius of the sphere is ρ.

$$U = \{(u,v) | -\frac{\pi}{2} \leq u \leq \frac{\pi}{2}, -\pi \leq v \leq \pi\} \tag{19.13}$$

Using trigonometry and the definition of spherical coordinates, we have:

$$c(u,v) = \begin{bmatrix} \rho\cos(u)\cos(v) \\ \rho\cos(u)\sin(v) \\ \rho\sin(u) \end{bmatrix} \tag{19.14}$$

We can differentiate to calculate c_u and c_v:

$$c_u = \begin{bmatrix} -\rho \sin u \cos v \\ -\rho \sin u \sin v \\ \rho \cos u \end{bmatrix}, c_v = \begin{bmatrix} -\rho \cos u \sin v \\ \rho \cos u \cos v \\ 0 \end{bmatrix} \tag{19.15}$$

Notice that ${c_u}^T c_v = 0$. Using the formulas above, we thus calculate K, M, and T as functions of u and v:

$$K = \begin{bmatrix} \frac{1}{\rho} & 0 \\ 0 & \frac{1}{\rho} \end{bmatrix}, M = \begin{bmatrix} \rho & 0 \\ 0 & \rho \cos u \end{bmatrix}, T = \begin{bmatrix} 0 & \frac{\tan v}{\rho} \end{bmatrix} \tag{19.16}$$

19.2.4 Velocity of Contact Points

For a given Gauss Frame, we can define the body velocity of any surface point relative to the fixed frame of the object. Let $\alpha(t)$ be an arbitrary coordinate pair (u, v). $c(\alpha(t))$ is the associated point surface point (relative to the fixed frame of the object). Thus, we can calculate $g_{oc}(t) \in \mathbb{R}^3$ and the components of V_{oc}^b as shown below. Note the use of the chain rule in calculating $\dot{x}$, $\dot{y}$, $\dot{z}$, and $\dot{p_{oc}}$. Remember also that $z = n$.

$$p_{oc}(t) = p(t) = c(\alpha(t)), R_{oc}(t) = [x(t), y(t), z(t)] = \begin{bmatrix} \frac{c_u}{\|c_u\|} & \frac{c_v}{\|c_v\|} & \frac{c_u \times c_u}{\|c_u \times c_v\|} \end{bmatrix}$$

$$v_{oc} = R_{oc}^T \dot{p}_{oc}(t) = \begin{bmatrix} x^T \\ y^T \\ z^T \end{bmatrix} \frac{\partial c}{\partial \alpha} \dot{\alpha} = \begin{bmatrix} x^T \\ y^T \\ z^T \end{bmatrix} \begin{bmatrix} c_u & c_v \end{bmatrix} \dot{\alpha} = \begin{bmatrix} \|c_u\| & 0 \\ 0 & \|c_v\| \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} M\dot{\alpha} \\ 0 \end{bmatrix}$$

$$\hat{\omega}_{oc} = R_{pc}^T \dot{R}_{oc} = \begin{bmatrix} x^T \\ y^T \\ z^T \end{bmatrix} \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} \end{bmatrix} = \begin{bmatrix} 0 & x^T \dot{y} & x^T \dot{z} \\ y^T \dot{x} & 0 & y^T \dot{z} \\ z^T \dot{x} & z^T \dot{y} & 0 \end{bmatrix}$$

$$y^T \dot{x} = y^T [x_u \; x_v] \dot{\alpha} = TM\dot{\alpha}$$

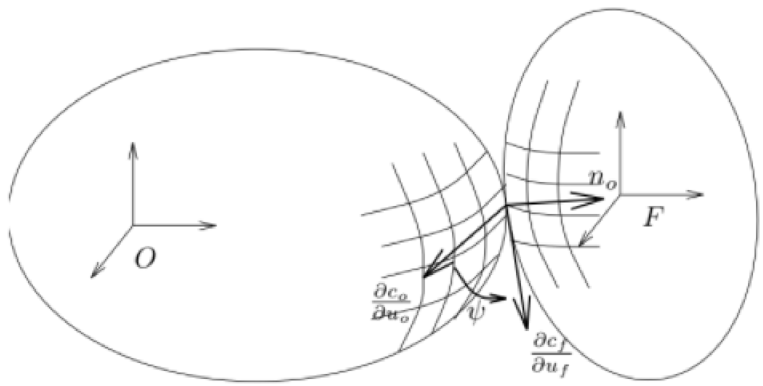
$$\begin{bmatrix} x^T \dot{z} \\ y^T \dot{z} \end{bmatrix} = \begin{bmatrix} x^T \\ y^T \end{bmatrix} \dot{z} = \begin{bmatrix} x^T \\ y^T \end{bmatrix} \begin{bmatrix} n_u & n_v \end{bmatrix} \dot{\alpha} = KM\dot{\alpha}$$

$$\hat{\omega}_{oc} = \left[\begin{array}{cc|c} 0 & -TM\dot{\alpha} & \multirow{2}{*}{$KM\dot{\alpha}$} \\ TM\dot{\alpha} & 0 & \\ \hline \multicolumn{2}{c|}{-(KM\dot{\alpha})^T} & 0 \end{array} \right]$$

$$v_{oc} = \begin{bmatrix} M\dot{\alpha} \\ 0 \end{bmatrix}$$

19.2.5 Contact Kinematics

For analyzing contacts, we define two Gauss frames, one for the finger (F) and one for the object (O). At the point of contact, the two Gauss frames' z axes are antiparallel. The **angle of contact**, the angle between their x axes, is ϕ (this is equivalent to the angle between their y axes).



Thus, we have:

$$R_{c_o c_f} = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ -\sin\phi & -\cos\phi & 0 \\ 0 & 0 & -1 \end{bmatrix}, p_{c_o c_f} = 0 \in \mathbb{R}^3 \tag{19.17}$$

Let $\alpha_o = (u_o, v_o)$ and $\alpha_f = (u_f, v_f)$, the contact points on each object in U_o and U_f respectively. We parameterize a contact as $\eta = (\alpha_o, \alpha_f, \phi)$.

19.2.6 Montana Equations of Contact

At any given contact point, the velocity can be parameterized by 5 free parameters: ω_x, ω_y, ω_z, v_x, and v_y. The velocity of the contact point into/out of either object, v_z, is 0 for rigid objects. ω_x and ω_y describe rolling velocities while v_x and v_y describe sliding velocities.

Let us first define the curvature of O relative to C_f as $\tilde{K}_o = R_\phi K_o R_\phi$, where

$$R_\phi = \begin{bmatrix} \cos\phi & -\sin\phi \\ -\sin\phi & -\cos\phi \end{bmatrix} \tag{19.18}$$

$K_f + \tilde{K}_o$ is termed relative curvature. Note that K, M, and T are defined independently as above for each Gauss Frame. Be aware also that ϕ and ψ are used interchangeably in the slides.

We can then derive the Montana Equations of Contact:

$$
\begin{aligned}
\dot{\alpha}_f &= {M_f}^{-1}(K_f + \tilde{K}_o)^{-1}(\begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} - \tilde{K}_o \begin{bmatrix} v_x \\ v_y \end{bmatrix}) \\
\dot{\alpha}_o &= {M_o}^{-1} R_\phi (K_f + \tilde{K}_o)^{-1}(\begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} + K_f \begin{bmatrix} v_x \\ v_y \end{bmatrix}) \\
\dot{\phi} &= \omega_z + T_f M_f \dot{\alpha}_f + T_o M_o \dot{\alpha}_o \\
v_z &= 0
\end{aligned} \tag{19.19}
$$

We can reformat the results of these equations into the form of a standard nonholonomic system, with ω_x and ω_y as the inputs:

$$
\begin{bmatrix} \dot{u}_f \\ \dot{v}_f \\ \dot{u}_o \\ \dot{v}_o \\ \dot{\phi} \end{bmatrix} = \dot{\eta} = g_1(\eta)\omega_x + g_2(\eta)\omega_y \tag{19.20}
$$

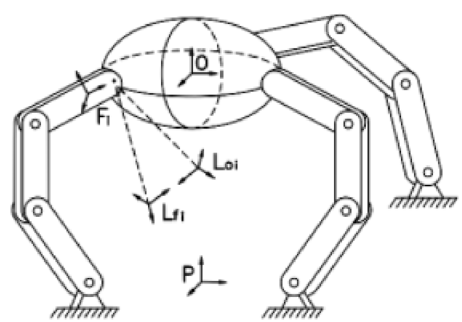
19.3 Introduction to Hand Kinematics

19.3.1 Hand Jacobian

Ideally, we want equations relating motions of the grasped object to motions of the hand. We can consider k fingers manipulating an object as k robots working on the same object O. The robot joint angles to end-effector position/velocity relationship is given by the Jacobian for each finger; we will combine these Jacobians into a Hand Jacobian for the full system. There is also the important constraint that contact must be maintained, i.e. the normal force must be positive.

We first use the coordinate frames of the fingers and contacts to derive a relationship between the velocity of the object and the velocities of the fingers.

$$
\begin{aligned}
g_{po} &= g_{pf_i}(\theta_i) g_{f_i l_{f_i}} g_{l_{f_i} l_{o_i}} g_{l_{o_i}} \\
V_{po} &= \mathrm{Ad}_{g_{f_i o}^{-1}} V_{pf_i} + \mathrm{Ad}_{g_{l_{o_i} o}^{-1}} V_{l_{f_i} l_{o_i}} \\
\mathrm{Ad}_{g_{l_{o_i} o}} V_{po} &= \mathrm{Ad}_{g_{f_i l_{o_i}}^{-1}} V_{pf_i} + V_{l_{f_i} l_{o_i}}
\end{aligned}
$$



For this example, we consider point contacts without friction (PCWF), i.e. $B_i = e_3 \in \mathbb{R}^{6\times 1}$. We might encounter PCWF in the case of flat finger pads interacting with round, convex objects. By the fourth Montana Equation of Contact, we have:

$$
V_z = 0 \Rightarrow [0\,0\,1\,0\,0\,0] V_{l_{f_i} l_{o_i}} = B_i^T V_{l_{f_i} l_{o_i}} = 0
$$

This leaves us with the following relation, where the velocities of the fingers can be calculated using their individual manipulator Jacobians. Note that, unconventionally, θ_i refers to the vector of joint angles for the ith finger, rather than the joint angle of the ith joint.

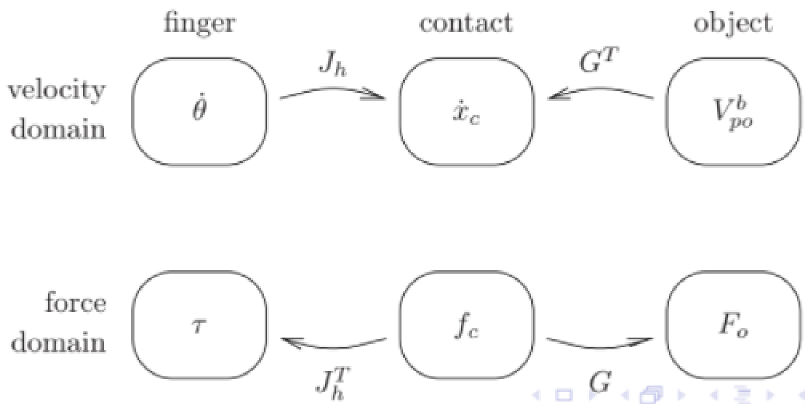
$$B_i^T \mathrm{Ad}_{g_{l_{o_i} o}} V_{po} = \mathrm{Ad}_{g_{f_i l_{o_i}}} \underbrace{V_{pf_i}}_{J_i(\theta_i)\dot{\theta}_i}$$

Stacking the relations for each finger into a single vectorized expression gives the following definition and interpretations/usages of the Hand Jacobian J_h:

$$\begin{bmatrix} B_1^T \mathrm{Ad}_{g_{l_{o_i} o}} \\ \vdots \\ B_k^T \mathrm{Ad}_{g_{l_{o_k} o}} \end{bmatrix} V_{po} = \begin{bmatrix} \mathrm{Ad}_{g^{-1}_{f_i l_{o_1}}} J_1(\theta_1) & & \\ & \ddots & \\ & & \mathrm{Ad}_{g^{-1}_{f_k l_{o_k}}} J_k(\theta_k) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_k \end{bmatrix}$$

$$\boxed{G^T(\eta) V_{po} = J_h(\theta, x_0, \eta)\dot{\theta}}$$

$$\theta = (\theta_1, \ldots, \theta_k) \in \mathbb{R}^n, n = \sum_{i=1}^{k} n_i, J_h \in \mathbb{R}^{m \times n}: \text{Hand Jacobian}$$



Knowing this Jacobian, we can define a **multifingered grasp** as $\Omega = (G, FC, J_h)$.

19.3.2 Alternative Methods

While this math is the solution in principle, it is incredibly complicated. As the number and complexity of fingers increases, it eventually may make sense to resort to deep learning methods.