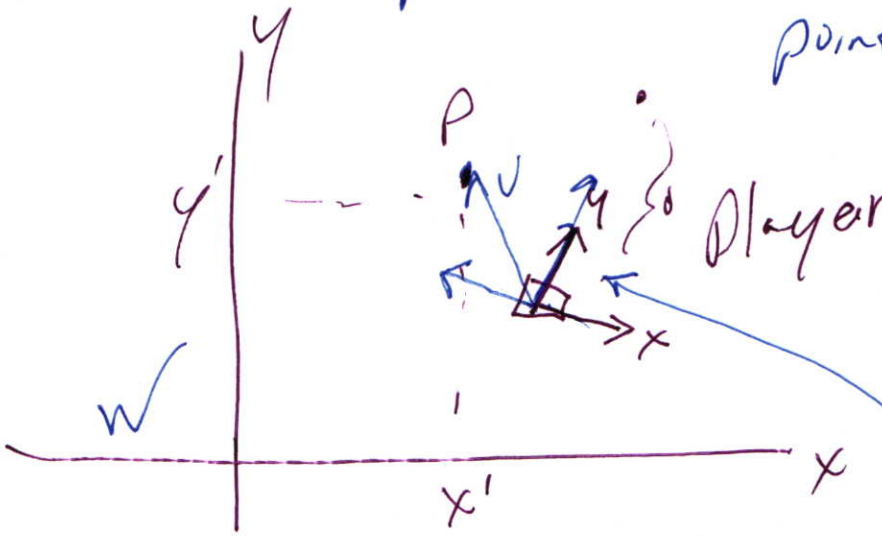
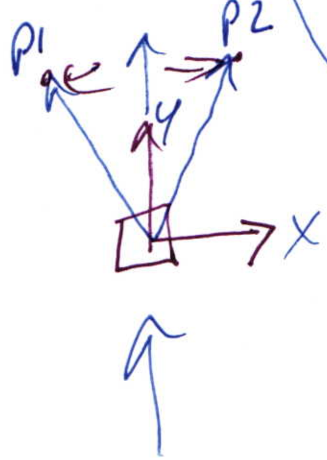
Lecture notes
Thursday Sept. 12

Problem: given vector v to point P in coordinate system W, where is P in local coordinate system defined by $\vec{y}$



where is point P in coordinate system of player

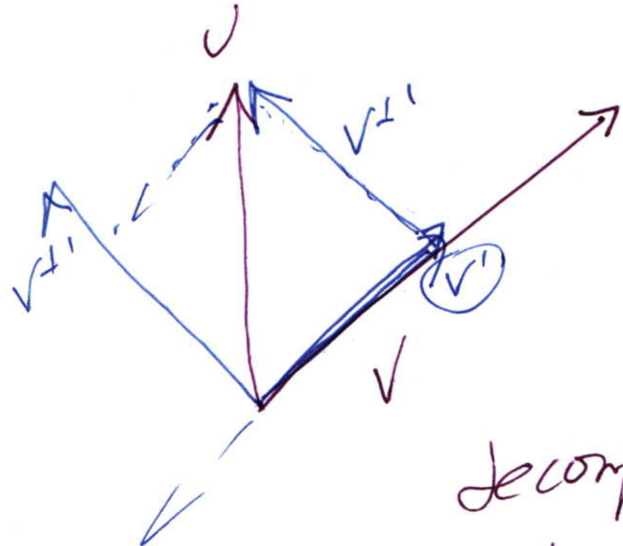


issue here

are points p1 and p2 left or right of y vector?

orthogonal projection
See Amant notes lecture 4

$u = v' + v^{\perp'}$



decomposition of u into vector v, $v^{\perp}$?

project u onto v?

$$v' = \frac{u \cdot v}{|v|}\left(\frac{v}{|v|}\right)$$

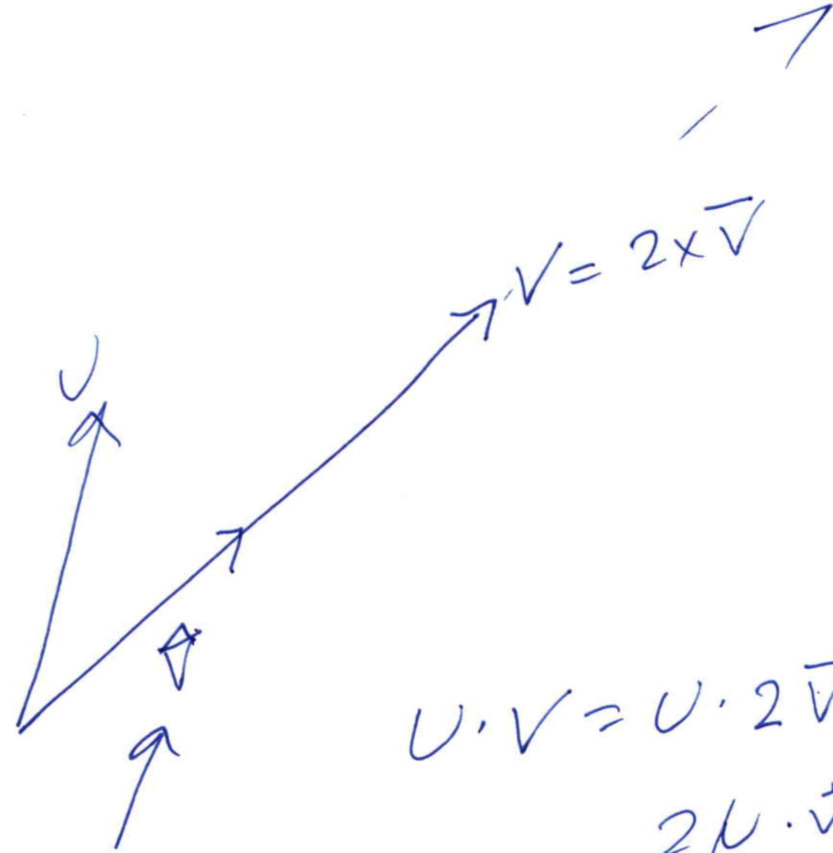
$v = \langle x, y \rangle$

$v^{\perp} = \langle -y, x \rangle$

$$\to v^{\perp'} = u - v'$$

$$\to v^{\perp'} = \frac{u \cdot v^{\perp}}{|v|}\left(\frac{v^{\perp}}{|v|}\right)$$

Dot product increases with
Length of vector



$U \cdot V = U \cdot 2\bar{V}$

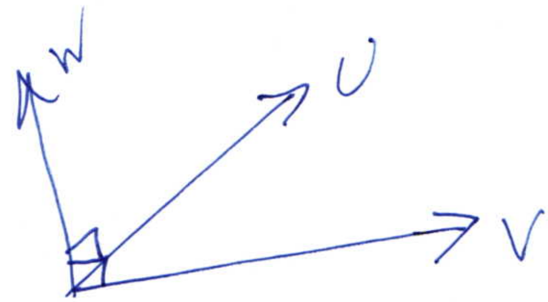
$2(U \cdot \vec{V})$

best to
use
normalized
vector

Cross product

vector W is cross product of U, V

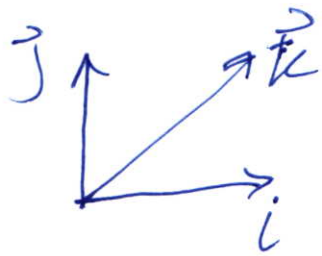
$W = U \times V$



$\left.\begin{matrix} W \perp V \\ W \perp U \end{matrix}\right]$ W is orthogonal to V, U

$U = \langle U_x, U_y, U_z \rangle \quad V = \langle V_x, V_y, V_z \rangle$

$$U \times V = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ U_x & U_y & U_z \\ V_x & V_y & V_z \end{vmatrix}$$



$i = \langle 1, 0, 0 \rangle$

$j = \langle 0, 1, 0 \rangle$

$k = \langle 0, 0, 1 \rangle$

$$\vec{U} \times \vec{V} = \vec{i}\,(U_y V_z - U_z V_y)$$
$$- j\,(U_x V_z - U_z V_x)$$
$$+ k\,(U_x V_y - U_y V_x)$$

page 5

example

$U = \langle 1, 5, 0 \rangle$

$V = \langle 0, 1, 3 \rangle$

$$U \times V = \begin{vmatrix} i & j & k \\ 1 & 5 & 0 \\ 0 & 1 & 3 \end{vmatrix}$$

$$= i(\underline{15}) - j\,\underline{3} + k\,\underline{1}$$

$$= \langle 15, -3, 1 \rangle$$

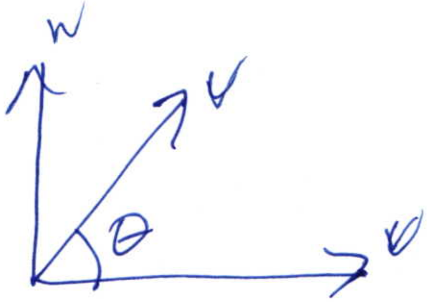
$|\vec{U}|, |\vec{V}| = 1$ normalized

$\vec{U} \times \vec{V} = \underline{1}$

$U \times V$ if $U \perp V$

$U \parallel V \quad V \times U = 0$

Sin rule for cross product



$$|u \times v| = |u||v| \sin\theta$$

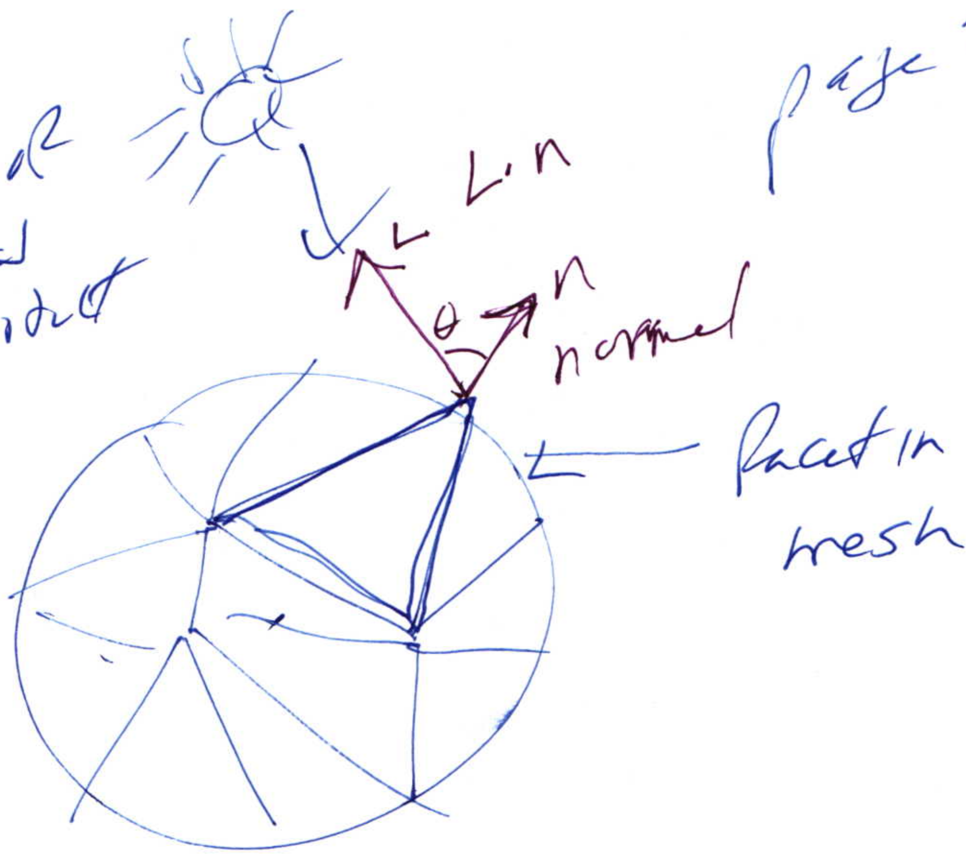
$$\theta = 90^\circ \Rightarrow 1$$

$$|u \times v| = |u||v|$$

$$\theta = 0^\circ \Rightarrow 0$$

$$|u \times v| = 0$$

application of cross and dot product

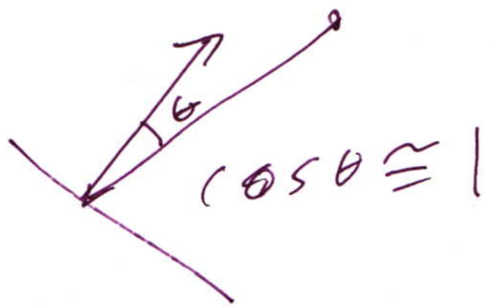


L·n larger

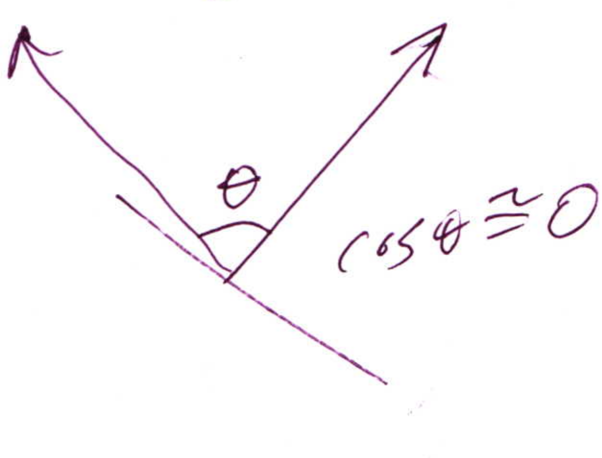
$$|L \cdot n| = |L|\,|n| \cos\theta$$

$|v|$ magnitude of v

small angle

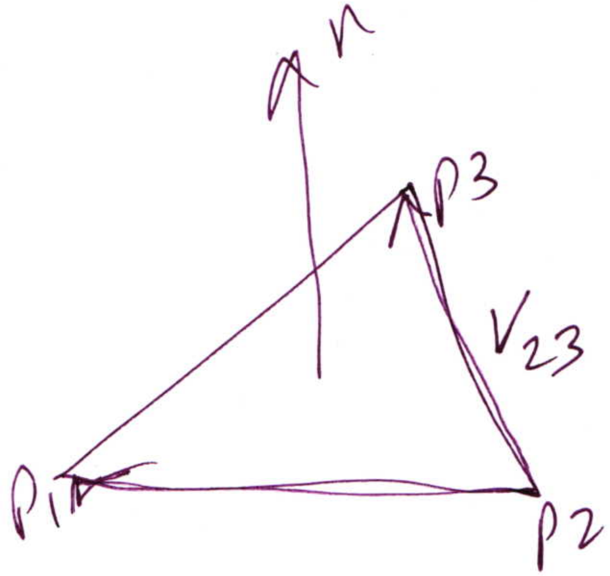


large angle



Computing cross product for triangle of 3 pts

p_1, p_2, p_3 page 8



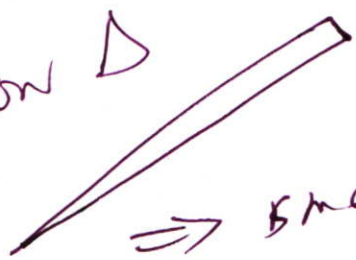
$V_{23} = P_3 - P_2$

$V_{21} = P_1 - P_2$

$$\hat{n} = \frac{V_{23} \times V_{21}}{|V_{23} \times V_{21}|}$$

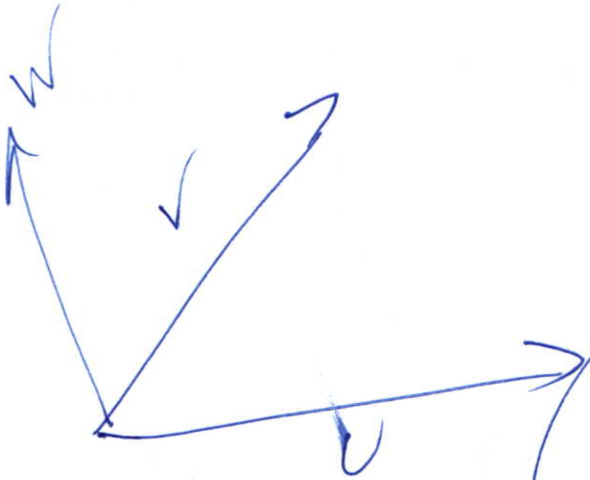
$\vec{n}$ $\hat{n}$ $\hat{n}$

narrow Δ



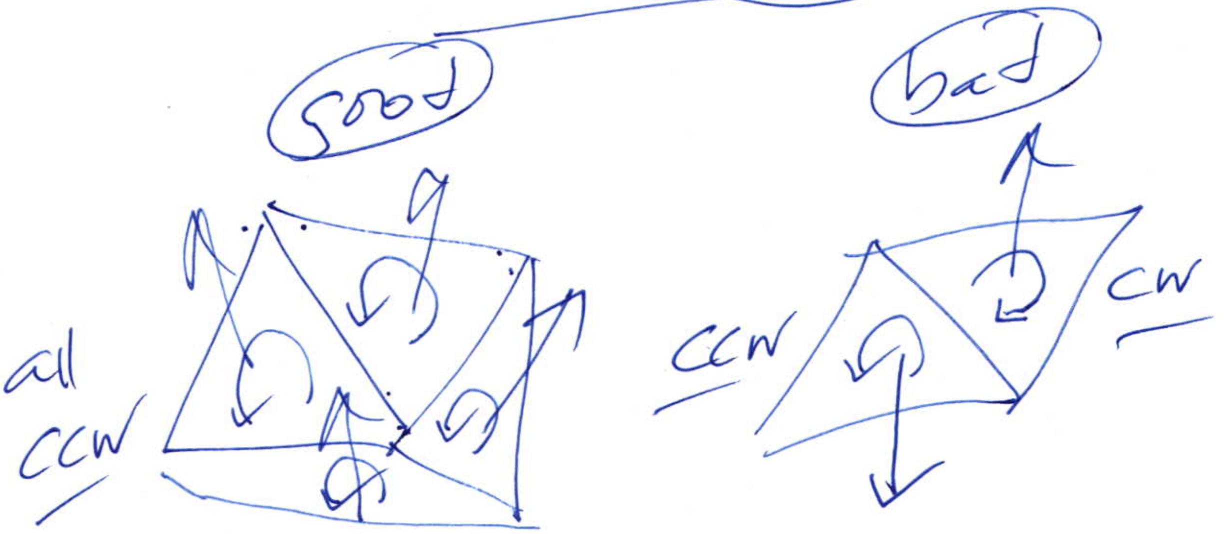
$\Rightarrow$ small θ $\Rightarrow$ noisy <u>normal</u>

cross product changes sign



~~v×w~~

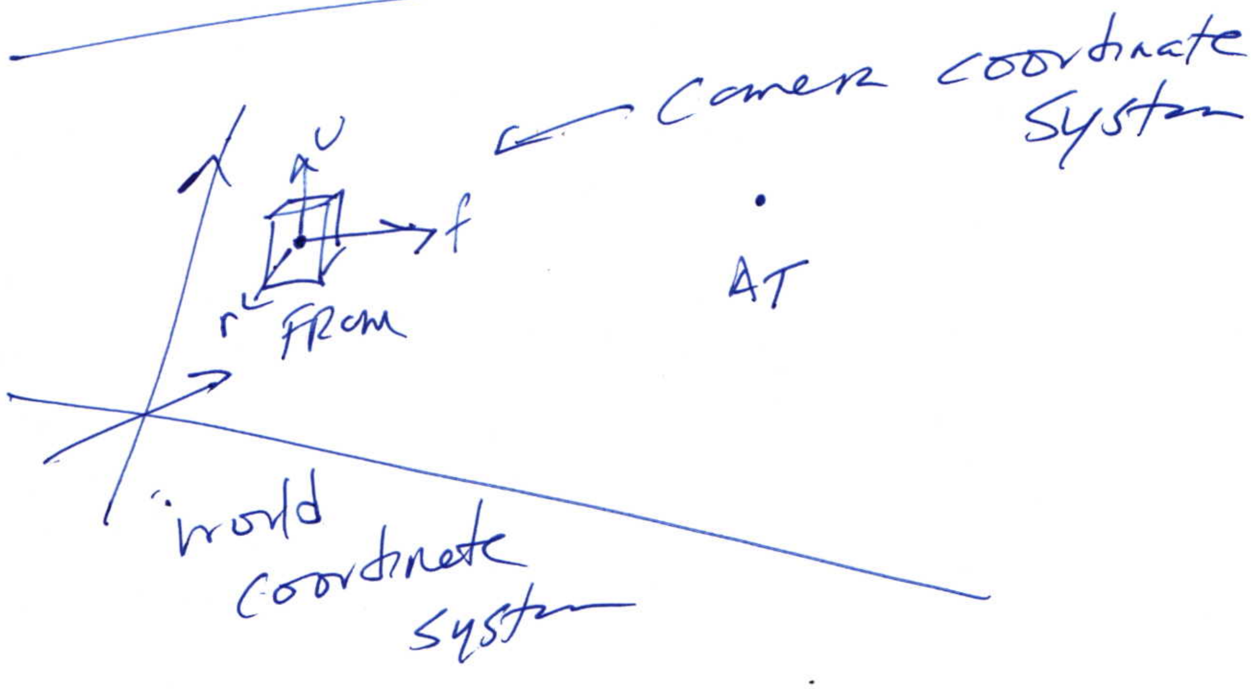
$u \times v = -v \times u$



Important that triangles in mesh go same direction, clockwise or counter cw, so cross product vector goes same direction

Tiny planet problem

⇒ camera problem



CMSC 425: Lecture 5
More on Geometry and Geometric Programming

More Geometric Programming: In this lecture we continue the discussion of basic geometric programming from the previous lecture. We will discuss the cross-product, orientation testing, and homogeneous coordinates.

Cross Product: The cross product is an important vector operation in 3-space. You are given two vectors and you want to find a third vector that is orthogonal to these two. This is handy in constructing coordinate frames with orthogonal bases. There is a nice operator in 3-space, which does this for us, called the *cross product*.

The cross product is usually defined in standard linear 3-space (since it applies to vectors, not points). So we will ignore the homogeneous coordinate here. Given two vectors in 3-space, $\vec{u}$ and $\vec{v}$, their *cross product* is defined as follows (see Fig. 1(a)):

$$\vec{u} \times \vec{v} = \begin{pmatrix} u_y v_z - u_z v_y \\ u_z v_x - u_x v_z \\ u_x v_y - u_y v_x \end{pmatrix}.$$

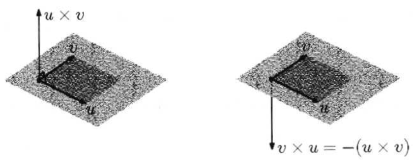


Fig. 1: Cross product.

A nice mnemonic device for remembering this formula, is to express it in terms of the following symbolic determinant:

$$\vec{u} \times \vec{v} = \begin{vmatrix} \vec{e}_x & \vec{e}_y & \vec{e}_z \\ u_x & u_y & u_z \\ v_x & v_y & v_z \end{vmatrix}.$$

Here $\vec{e}_x$, $\vec{e}_y$, and $\vec{e}_z$ are the three coordinate unit vectors for the standard basis. Note that the cross product is only defined for a pair of free vectors and only in 3-space. Furthermore, we ignore the homogeneous coordinate here. The cross product has the following important properties:

Skew symmetric: $\vec{u} \times \vec{v} = -(\vec{v} \times \vec{u})$ (see Fig. 3(b)). It follows immediately that $\vec{u} \times \vec{u} = 0$ (since it is equal to its own negation).

Nonassociative: Unlike most other products that arise in algebra, the cross product is *not* associative. That is

$$(\vec{u} \times \vec{v}) \times \vec{w} \neq \vec{u} \times (\vec{v} \times \vec{w}).$$

Bilinear: The cross product is linear in both arguments. For example:

$$\begin{aligned} \vec{u} \times (\alpha\vec{v}) &= \alpha(\vec{u} \times \vec{v}), \\ \vec{u} \times (\vec{v} + \vec{w}) &= (\vec{u} \times \vec{v}) + (\vec{u} \times \vec{w}). \end{aligned}$$

Perpendicular: If $\vec{u}$ and $\vec{v}$ are not linearly dependent, then $\vec{u} \times \vec{v}$ is perpendicular to $\vec{u}$ and $\vec{v}$, and is directed according the right-hand rule.

Angle and Area: The length of the cross product vector is related to the lengths of and angle between the vectors. In particular:

$$|\vec{u} \times \vec{v}| = |u||v| \sin\theta,$$

where θ is the angle between $\vec{u}$ and $\vec{v}$. The cross product is usually not used for computing angles because the dot product can be used to compute the cosine of the angle (in any dimension) and it can be computed more efficiently. This length is also equal to the area of the parallelogram whose sides are given by $\vec{u}$ and $\vec{v}$. This is often useful.

The cross product is commonly used in computer graphics for generating coordinate frames. Given two basis vectors for a frame, it is useful to generate a third vector that is orthogonal to the first two. The cross product does exactly this. It is also useful for generating surface normals. Given two tangent vectors for a surface, the cross product generate a vector that is normal to the surface.

Example–Tiny-planet frame: Inspired by tiny-planet photos (see Fig. 2(a)), let us consider how to construct a local coordinate system for a player object standing on the sphere. Let c denote the sphere's center point, and let p denote the point on the sphere where the player object is standing (see Fig. 2(b)). In order to indicate the direction in which the player is facing, a second point $q \neq p$ is given on the surface of the sphere. These two points define a great-circle on the sphere. The player's *up axis* $\vec{u}$ is directed along a ray from c through p, the *forward axis* $\vec{f}$ is tangent to the minor great-circle arc from p to q, and the *right axis* $\vec{r}$ is orthogonal to these two and is directed to the player's right (see Fig. 2(c)).

Question: Given c, p, and q, how can we construct the vectors $\vec{u}$, $\vec{f}$, and $\vec{r}$ of the player's local coordinate frame?

Answer: First, the player's up-vector $\vec{u}$ is just the normalization of the vector from c through p, that is

$$\vec{u} \leftarrow \text{normalize}(p - c) = \frac{p - c}{\|p - c\|}.$$

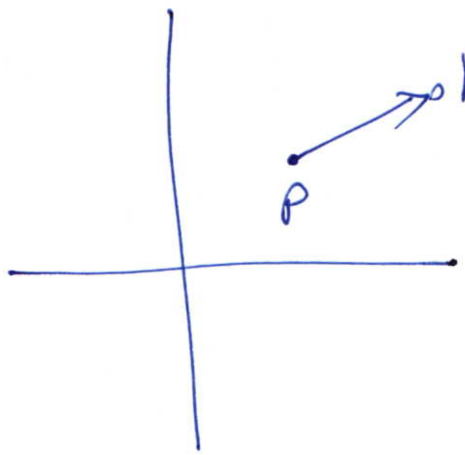
(Recall that the length of a vector $\vec{v}$ can be computed as $\|\vec{v}\| \leftarrow \sqrt{\vec{v} \cdot \vec{v}}$.)

Next, to compute the player's right-vector $\vec{r}$, we observe that it must be perpendicular to the equatorial plane containing p and q, or equivalently, it must be perpendicular to both of the up-vector $\vec{u}$ and $\vec{w} = q - c$. Using the standard right-handed cross product, we have

$$\vec{r} \leftarrow \text{normalize}(\vec{w} \times \vec{u}), \quad \text{where } \vec{w} \leftarrow q - c.$$

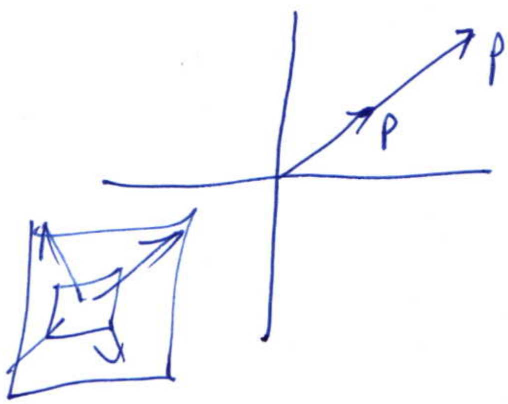
You might wonder why $\vec{r}$ is not coming out of c. Recall that these are *free* vectors, and hence they are not associated with any particular location in space. (Note that the

Translations and scalings



translate

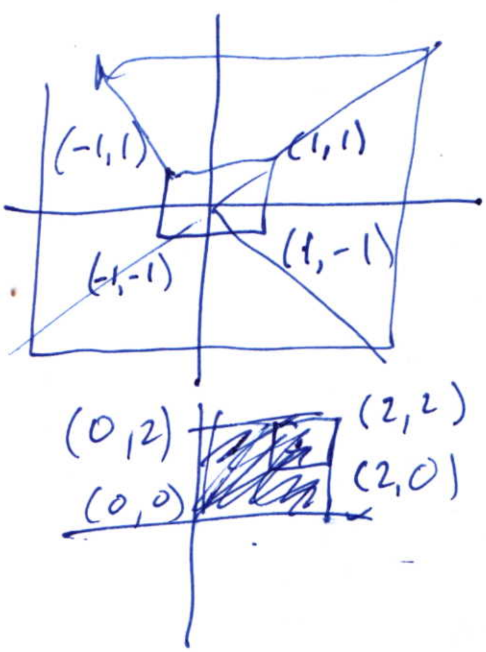
$p' = p + T_{<3,4>}$



scale

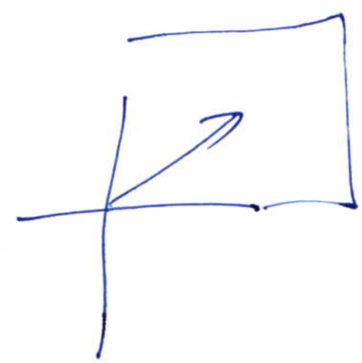
$p' = \cancel{2}\, p$

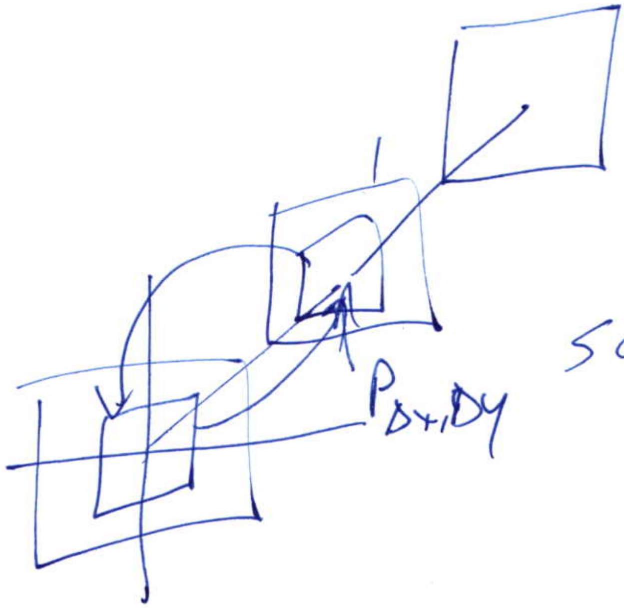
$p' = 2p$



P_1, P_2, P_3, P_4

$\times 2$





scaling $\Rightarrow$ translation

$$T = \langle \Delta x, \Delta y \rangle$$

$$Shape' = \alpha(Shape - T) + T$$

$$= \alpha_1(\alpha_1(Shape - T_1) + T_1) - T_2) + T_2$$

Complicated expressions with sequence of translations and scalings.

Homogeneous coordinates

$P = (x, y, w)$ — 3rd coordinate is homogeneous coordinate

two d points

$P_1 = (2, 3, 1)$ — $w = 1$ usually 1.

$P_2 = (-5, 6, 1)$

$$v = P_2 - P_1 = \langle -7, 3, 0 \rangle$$

$$v_1 + v_2 = \langle v_{x_1} + v_{x_2}, v_{y_1} + v_{y_2}, 0 \rangle$$

$$P_1 + P_2 = (-3, 9, \underbrace{1+1}_{2})$$

points - $w = 1$
vectors - $w = 0$

add two pts get $w = 2$, not good
subtract two pts, get vector w/ $w = 0$

Combining translations and scalings in homogeneous coordinates using homogeneous matrices 14

$$M_S * P = \begin{bmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} \alpha x \\ \alpha y \\ 1 \end{bmatrix}$$

$$M_T * P = \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \cdot x + \Delta x \cdot 1 \\ y + \Delta y \\ 1 \end{bmatrix}$$

$$M_T * M_S * M_T (P)$$