Assignment 2: UAV Adaptive Path Following

Adaptive and Predictive Control [SC42040]

Ilario Azzollini, Kostantinos Kokkalis, Muhammad Ridho Rosa

March 16, 2017

Outline

Outline

2D Line Following

UAV Kinematics in 2D

$$\dot{x} = V_a cos(\psi) + W cos(\psi_w) + A cos(\psi_A)$$
$$\dot{y} = V_a sin(\psi) + W sin(\psi_w) + A sin(\psi_A)$$

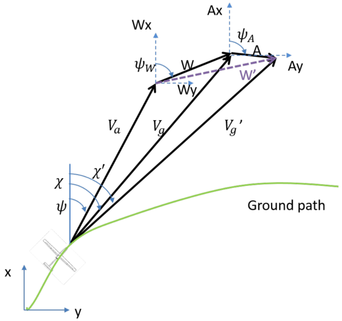


Figure: UAV Kinematics

- x,y: position of UAV
- V_a: airspeed of UAV
- ψ: heading angle between airspeed and horizontal axis
- χ': UAV course angle
- W: constant wind amplitude
- ψ_w: angle of constant wind in earth frame
- A: time varying wind amplitude
- ψ_A: angle of time varying wind in earth frame

2D Line Following

The following assumptions are made:

- altitude and airspeed are held constant by the longitudinal control of the UAV
- the UAV is equipped with the course-hold loop devices whose dynamics can be modeled as a first-order system
- the UAV course is measurable
- the slowly time-varying unknown component of the wind has amplitude A and angle ψ_A

2D Line following

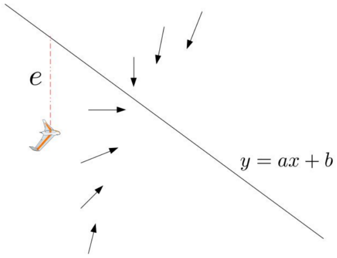


Figure: Vector Field straight line following schematic

Dynamics and Tracking Errors

Course dynamics:

$$\dot{\chi}' = \alpha(\chi_c - \chi')$$

Distance error:

$$e = y - (ax + b)$$

Course error:

$$\tilde{\chi}' = \chi' - \chi_d$$

Desired course:

$$\chi_d = -\chi^{\infty}\frac{2}{\pi}tan^{-1}(ke) + tan^{-1}(a)$$

2D Line Following

Command course

$$\chi_c = \chi' - \frac{1}{\alpha}\chi^{\infty}\frac{2}{\pi}\frac{k}{1+(ke)^2}\hat{V_g}'(\sin\chi' - a\cos\chi') - \frac{\kappa}{\alpha}sat\left(\frac{\tilde{\chi}'}{\epsilon}\right)$$

Ground velocity estimator

$$\dot{\hat{V}}_g' = \frac{\partial V_g'}{\partial \chi'}\left[-\chi^{\infty}\frac{2}{\pi}\frac{k}{1+(ke)^2}(\sin\chi' - a\cos\chi')\hat{V_g}' - \kappa sat\left(\frac{\tilde{\chi}'}{\epsilon}\right)\right]\cdot$$
$$\cdot\,\Gamma\rho\tilde{\chi}'\chi^{\infty}\frac{2}{\pi}\frac{k}{1+(ke)^2}(\sin\chi' - a\cos\chi')$$

Ground velocity estimate partial derivative

$$\frac{\partial V_g{}'}{\partial \chi'} \approx W\sin(\psi_w - \chi') + [V_a^2 - W^2\sin^2(\psi_w - \chi')]^{-\frac{1}{2}}W^2\sin(\psi_w - \chi')\cos(\psi_w - \chi')$$

Lyapunov stability arguments are used to demonstrate asymptotic decay of path-following and estimation errors by Bingyu Zhou, "Adaptive Path Following for UAV in Time Varying Unknown Wind Environments" lecture slides, *Adaptive and Predictive Control 2017*

2D Line Following

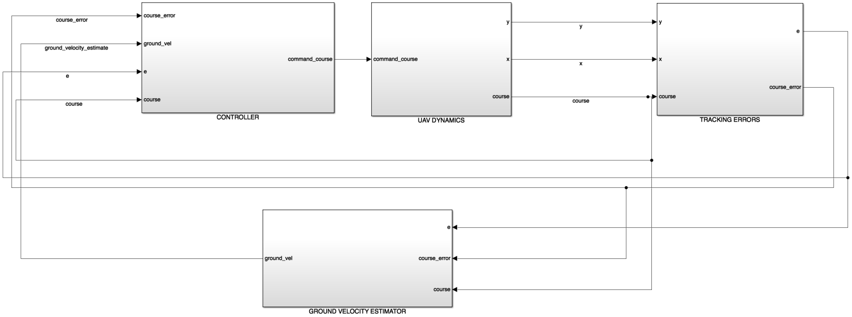


Figure: Simulink scheme

2D Line Following

Parameter	Value
W	6
ψ_w	$(230/180)\pi$
V_a	13
A	3
ψ_A	π
χ^∞	$\pi/2$
α	1.65
k	0.1
κ	$\pi/2$
ϵ	0.5
Γ	50
ρ	1
a	0
b	0

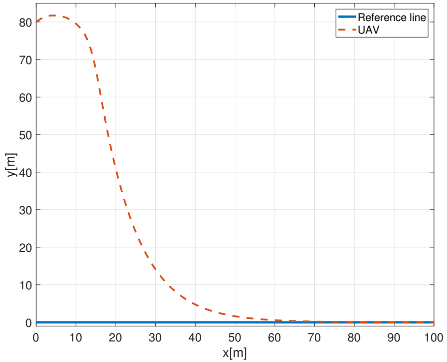


Figure: Line following for $y = 0x + 0$ starting from $x = 0, y = 80, \chi' = \pi/4$

Outline

2D Orbit Following

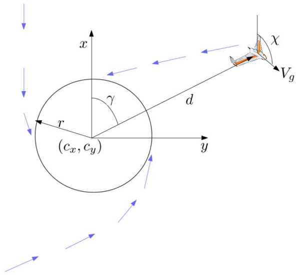


Figure: UAV Kinematics in polar coordinates

UAV Kinematics in Polar Coordinates

$$\dot{d} = V_g' \cos(\chi' - \gamma)$$

$$\dot{\gamma} = \frac{V_g'}{d} \sin(\chi' - \gamma)$$

Tracking Errors

Distance error:

$$\tilde{d} = d - r$$

Course error:

$$\tilde{\chi}' = \chi' - \chi_d$$

Desired course:

$$\chi_d = \gamma - \left[\frac{\pi}{2} + tan^{-1}(k\tilde{d})\right]$$

2D Orbit Following

Command course

$$\chi_c = \chi' + \frac{\hat{V}_g{}'}{\alpha d}\sin(\chi - \gamma) - \frac{\beta}{\alpha}\hat{V}_g{}'\cos(\chi' - \gamma) - \frac{\kappa}{\alpha} sat\left(\frac{\tilde{\chi}'}{\epsilon}\right)$$

Ground velocity estimator

$$\dot{\hat{V}}_g' = \frac{\partial V_g{}'}{\partial \chi'}\left(\frac{\hat{V}_g{}'}{d}\sin(\chi' - \gamma) - \beta\cos(\chi' - \gamma) - \kappa sat\left(\frac{\tilde{\chi}'}{\epsilon}\right)\right) + \\ - \Gamma\rho\tilde{\chi}'\left(\frac{\sin(\chi' - \gamma)}{d} - \beta\cos(\chi' - \gamma)\right)$$

Ground velocity estimate partial derivative

$$\frac{\partial V_g{}'}{\partial \chi'} \approx W\sin(\psi_w - \chi') + [V_a^2 - W^2\sin^2(\psi_w - \chi')]^{-\frac{1}{2}} W^2 \sin(\psi_w - \chi')\cos(\psi_w - \chi')$$

Lyapunov stability arguments are used to demonstrate asymptotic boundedness of path-following errors by us (first assignment) and Bingyu Zhou (lecture slides)

2D Orbit Following

Parameter	Value
initial d	15
initial χ'	$-\pi/4$
initial γ	$\pi/4$
cx	50
cy	50
R	10

The other parameters are the same already presented for the line following case

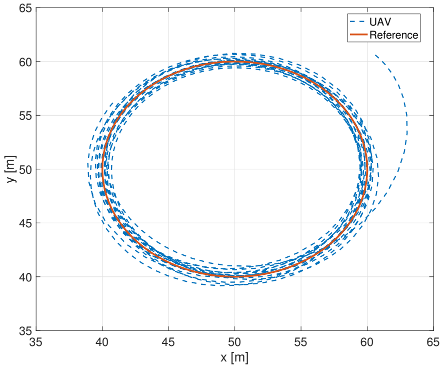


Figure: Orbit following example

Outline

Problem statement

Problems to be solved

- How to follow a line in the x-decreasing direction?
- How to follow an orbit in anticlockwise direction?
- Given two consecutive waypoints, how to switch from a line following to an orbit following task and vice versa?

Line Following problem

User problem

As a user, it would be preferable to give as input the x, y coordinates of the waypoints to be followed, instead of the a and b parameters of the line equation $y = ax + b$
This information can be used also to choose the direction in which the UAV will follow the line

Algorithm example

$$x_0 = 0;$$

$$y_0 = 0;$$

$$x_1 = -300;$$

$$y_1 = 0;$$

$$a = (y_1 - y_0)/(x_1 - x_0);$$

$$b = -(a * x_0) + y_0;$$

Line Desired Courses

Desired Courses

x-increasing direction:

$$\chi_d = -\chi^{\infty}\frac{2}{\pi}tan^{-1}(ke)+tan^{-1}(a)$$

x-decreasing direction:

$$\chi_d = \chi^{\infty}\frac{2}{\pi}tan^{-1}(ke) + tan^{-1}(a)$$

Remark

Use *atan2* instead of *atan*

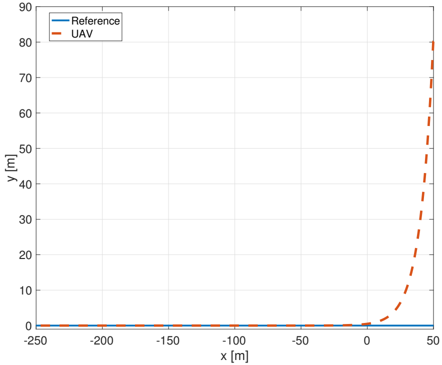


Figure: Same line following as before, but followed in the x-decreasing direction

The two-argument function $atan2(y,x)$

$atan2$ function gives a result $\varphi \in [-\pi, \pi]$ and it is defined as:

$$atan2(y,x) = \begin{cases} atan(\frac{y}{x}) & \text{if } x > 0 \\ atan(\frac{y}{x}) + \pi & \text{if } x < 0,\ y \geq 0 \\ atan(\frac{y}{x}) - \pi & \text{if } x < 0,\ y < 0 \\ \frac{1}{2}\pi & \text{if } x = 0,\ y > 0 \\ -\frac{1}{2}\pi & \text{if } x = 0,\ y < 0 \\ 0 & \text{if } x = 0,\ y = 0 \end{cases}$$

$atan(\frac{y}{x})$ gives a different (and **wrong**) angle with respect to $atan2(y,x)$ when $x < 0$.
In our case, for the desired course computation, we should use

$$atan2((y_1 - y_0),(x_1 - x_0))$$

D. Galli, "Vectors representation" lecture slides, *Physics 1, University of Bologna, 2017*

Orbit Desired Courses

Desired Courses

Clockwise direction:

$$\chi_d = \gamma - \left[\frac{\pi}{2} + tan^{-1}(k\tilde{d})\right]$$

Anticlockwise direction:

$$\chi_d = \gamma + \left[\frac{\pi}{2} + tan^{-1}(k\tilde{d})\right]$$

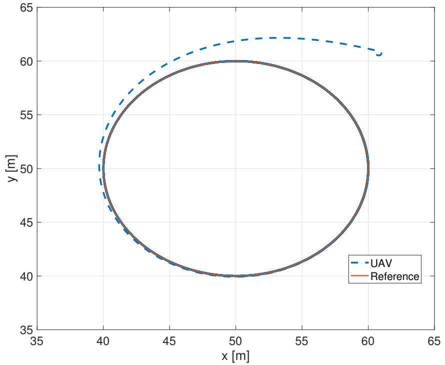


Figure: Same orbit following as before, but followed in anticlockwise direction

Stopping criteria

Last problem to be solved

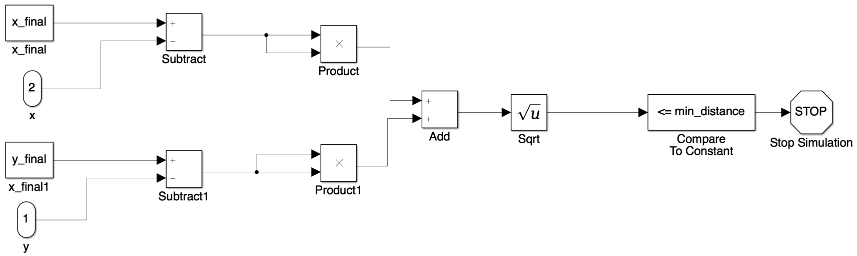
When to stop the line following?
Intuitively, when the UAV is at a certain distance from the next waypoint. Now that our algorithm takes as input the waypoints coordinates x, y, this is really easy to be implemented.

What about a stopping criterion for the orbit following?
Less intuitive. Think about the fact that a line intersects a circumference in 2 points. You can obtain these points in Matlab by using the command *linecirc*

Stopping criteria

Stopping criteria

In both cases (line and orbit) we want to stop the simulation when the UAV is close enough to a target point.
For the *line*, the target point is the next waypoint.
For the *orbit*, we can compute the 2 points in which the **next** line to be followed intersects the orbit. The point that has the minimum distance from the next waypoint, will be our target point.

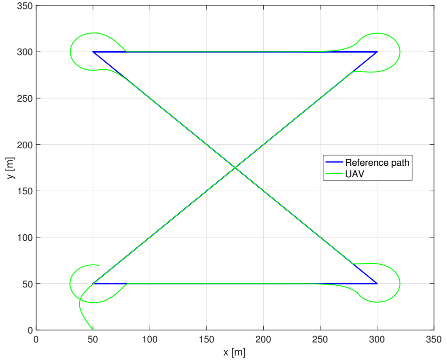


Outline

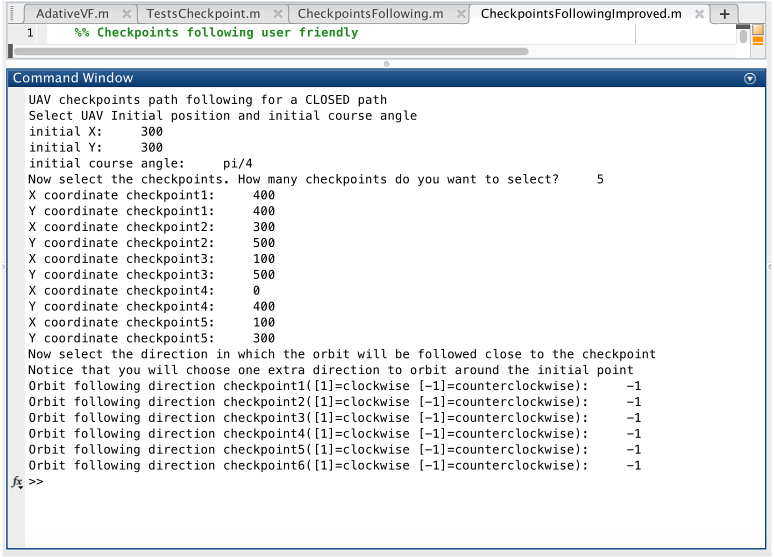
Waypoints following

Now we have all the tools for following a complex path combining line and orbit following. A Matlab script that consecutively uses the line following algorithm and the orbit following algorithm can be implemented.

Parameter	Value
$x_{initial}$	50
$y_{initial}$	0
$\chi'_{initial}$	$\pi/4$
(x_1, y_1)	$(300, 300)$
(x_2, y_2)	$(50, 300)$
(x_3, y_3)	$(300, 50)$
(x_4, y_4)	$(50, 50)$



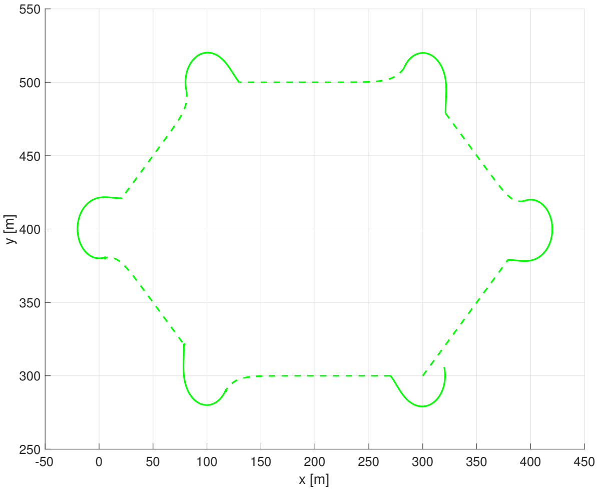
Basic user interface



```
%% Checkpoints following user friendly
```

```
UAV checkpoints path following for a CLOSED path
Select UAV Initial position and initial course angle
initial X:     300
initial Y:     300
initial course angle:     pi/4
Now select the checkpoints. How many checkpoints do you want to select?     5
X coordinate checkpoint1:     400
Y coordinate checkpoint1:     400
X coordinate checkpoint2:     300
Y coordinate checkpoint2:     500
X coordinate checkpoint3:     100
Y coordinate checkpoint3:     500
X coordinate checkpoint4:     0
Y coordinate checkpoint4:     400
X coordinate checkpoint5:     100
Y coordinate checkpoint5:     300
Now select the direction in which the orbit will be followed close to the checkpoint
Notice that you will choose one extra direction to orbit around the initial point
Orbit following direction checkpoint1([1]=clockwise [-1]=counterclockwise):     -1
Orbit following direction checkpoint2([1]=clockwise [-1]=counterclockwise):     -1
Orbit following direction checkpoint3([1]=clockwise [-1]=counterclockwise):     -1
Orbit following direction checkpoint4([1]=clockwise [-1]=counterclockwise):     -1
Orbit following direction checkpoint5([1]=clockwise [-1]=counterclockwise):     -1
Orbit following direction checkpoint6([1]=clockwise [-1]=counterclockwise):     -1
>>
```

Basic user interface

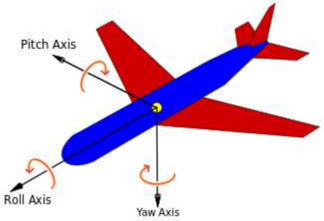


Outline

3D Line Following

UAV Kinematics in 3D

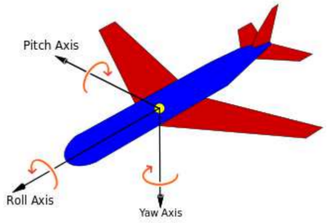
$$\dot{x} = V_a \cos\psi \cos\alpha_v + W \cos\psi_w \cos\alpha_w + A \cos\psi_A \cos\alpha_A$$
$$\dot{y} = V_a \sin\psi \cos\alpha_v + W \cos\alpha_w \sin\psi_w + A \cos\alpha_A \sin\psi_A$$
$$\dot{z} = V_a \sin\alpha_v + W \sin\alpha_w + A \sin\alpha_A$$



- x axis: Roll axis
- y axis: Pitch axis
- z axis: Yaw axis
  Notice in this way the frame is right-handed
- α_v, α_w, α_A: angles between the xy plane and the V_a, W and A vectors respectively (α_v is called in literature *flight path angle*)
- ψ, ψ_w, ψ_A: same angles as before

J.A. Guerrero, J.A. Escareno, Y. Bestaoui, "Quad-rotor MAV Trajectory Planning in Wind Fields", *IEEE International Conference on Robotics and Automation (ICRA) Karlsruhe, Germany, May 6-10, 2013*

3D Line Following



- The course angle of the UAV in the *xy* plane that we defined in 2D is practically a rotation about the *z* axis, so we will call it yaw course χ_y from now on
- Now, another course is needed in order to make the UAV able to reach a certain height: the pitch course χ_p
- Moreover, we consider the assumption that the UAV will not roll. In reality, also a roll control should be implemented by considering a roll course error

3D Line Following

Problem solution: Splitting the 3D problem into two 2D problems

- Control the yaw and the pitch courses separately, in order to properly orient the UAV in space and follow a certain line
- Project the line in space on the *xy* plane and use these informations for controlling the yaw course
- Project the line in space on the *yz* plane for controlling the pitch course

Course Dynamics

$$\dot{\chi}_y = \alpha(\chi_{cy} - \chi_y)$$
$$\dot{\chi}_p = \beta(\chi_{cp} - \chi_p)$$

3D Line Following

Tracking Errors

Distance errors:

$$e_{xy} = y - (a_{xy}x + b_{yx})$$
$$e_{yz} = z - (a_{yz}y + b_{yz})$$

Course errors:

$$\tilde{\chi}_y = \chi_y - \chi_{dy}$$
$$\tilde{\chi}_p = \chi_p - \chi_{dp}$$

Desired courses:

$$\chi_{dy} = \pm\chi^{\infty}\frac{2}{\pi}\tan^{-1}(ke_{xy}) + \tan^{-1}(a_{xy})$$
$$\chi_{dp} = \pm\chi^{\infty}\frac{2}{\pi}\tan^{-1}(ke_{yz}) + \tan^{-1}(a_{yz})$$

3D Line Following

The goal is then to find the control courses χ_{cy} and χ_{cp} that will drive all the errors to zero. The greatest advantage of approaching the problem in this way is that we already solved the problem in 2D, therefore we already proved the results.

Command courses

$$\chi_{cy} = \chi_y - \frac{1}{\alpha}\chi^{\infty}\frac{2}{\pi}\frac{k}{1+(ke_{xy})^2}\hat{V}'_{g,xy}(\sin(\chi_y) - a\cos(\chi_y)) - \frac{\kappa}{\alpha}sat(\frac{\tilde{\chi}_y}{\epsilon})$$
$$\chi_{cp} = \chi_p - \frac{1}{\beta}\chi^{\infty}\frac{2}{\pi}\frac{k}{1+(ke_{yz})^2}\hat{V}'_{g,yz}(\sin(\chi_p) - a\cos(\chi_p)) - \frac{\kappa}{\alpha}sat(\frac{\tilde{\chi}_p}{\epsilon})$$

Notice that we will also have two ground velocity estimators.

Outline

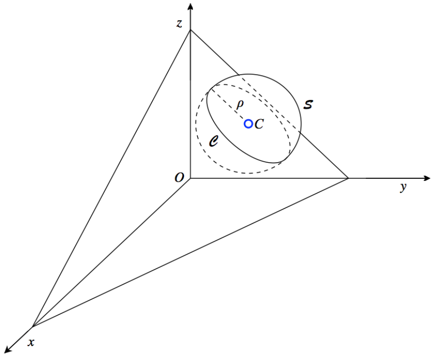
3D Orbit Following

Problem statement

Following a sphere in space would not be of great interest for implementing a 3D waypoints following, so we want to follow a circumference in space

Circumference in space: can be defined as the intersection of a plane and a sphere

$$\begin{cases} ax + by + cz = d \\ (x - x_C)^2 + (y - y_C) + (z - z_C) = \rho^2 \end{cases}$$



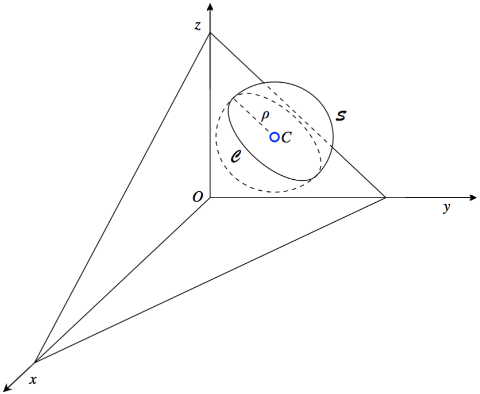
3D Orbit Following

Problem statement

The plane on which the circumference to be followed lies, in general will be parallel neither to the *xy* plane nor to the *yz* plane

Solution

Why not changing coordinates and work on a frame where the plane on which the circumference lies is parallel to the *xy* plane? In this way we could again split the problem in two: we will have a circumference in 2D to follow on the *xy* plane and a line to follow on the *yz* plane

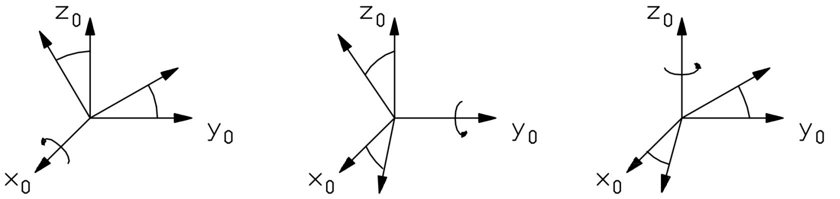


Coordinate change by means of Elementary Rotation Matrices

We can use elementary rotation matrices to change reference frame and apply this rotations to the vectors V_a, W and A. Then we work in this new reference frame where we will have **always** to follow an orbit in 2D on xy and a line in 2D on yz.

Elementary rotation matrices

Among all the possible rotations, there are 3 of them that are of interest: the 3 *elementary rotations*. They are rotations only about one of the 3 axes.
If we know the expressions of these 3 elementary rotations, we can express the rotation about any direction in space.



For more info about homogeneous transformations in general:
`http://www-lar.deis.unibo.it/people/cmelchiorri/Files_Robotica/FIR_03_Rbody.pdf`