Autonomous docking of two marine robots: a vector field based approach

Auguste Bourgois (PhD), Pr. Luc Jaulin

MOQESM 2018, Brest, SeaTechWeek
October 8-9-10, 2018

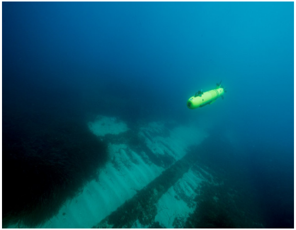
Overview I

Introduction

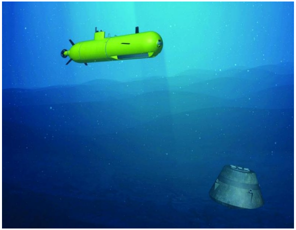
Examples of AUV operations

AUVs can be equipped with various types of sensors to perform different missions:

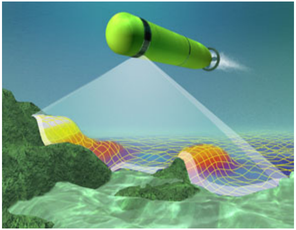
- Cameras for visual inspections
- Multibeam sonar for acoustic mapping/detection
- Magnetometers for magnetic mapping/detection



(a) ECA AUV inspecting a pipeline



(b) Thales AUV looking for underwater mines



(c) MBARI AUV mapping seafloor

AUV limitations

Figure: AUV deployment from a surface vessel (Courtesy of Subsea World News)

- Cost/duration of deployment/recovering of the AUV
- Limited battery life
- Limited storage capacity

Forssea's Remora

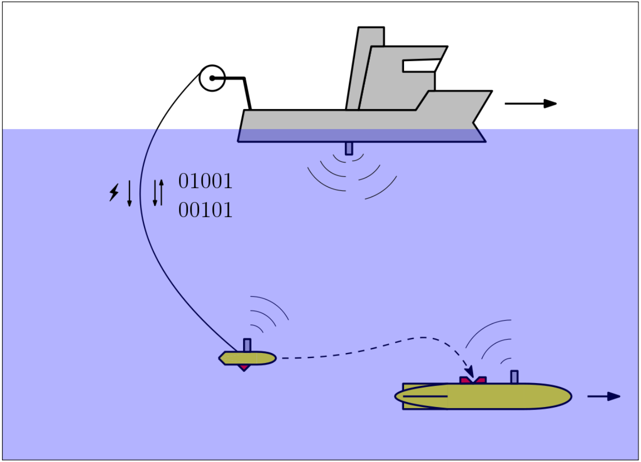


Figure: Autonomous dynamic docking of an ROV and an AUV

Problems from a control engineer's point of view

- The tether influence on the ROV's trajectory is unknown and quite unpredictable
- The targeted AUV is moving
- The ROV ought not crash onto fragile parts of the AUV
- The ROV ought not tie knots with its tether

Vector field based approach

Time-dependent attractive vector field

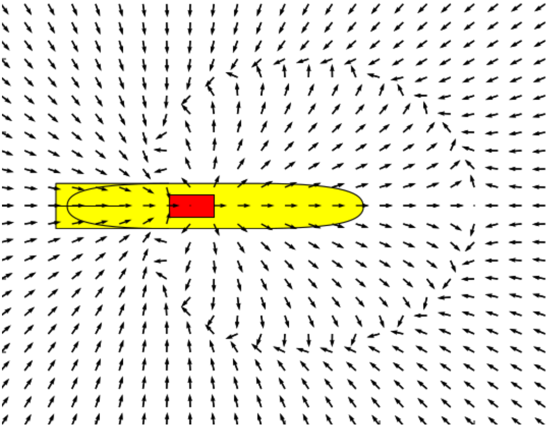


Figure: Attractive field generated by the targeted AUV

Method inspired from [Le Gallic et al., 2018], adapted for time-dependent vector fields.

Problem formalisation: frames

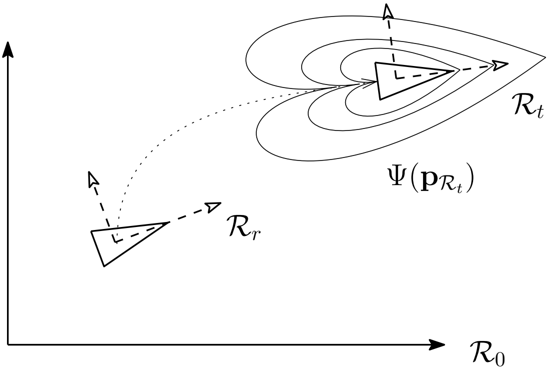


Figure: ROV, AUV, vector field and related frames

$$\boldsymbol{\Psi} : \mathbb{R}^3 \to \mathbb{R}^3$$
$$\mathbf{p} \mapsto \boldsymbol{\Psi}(\mathbf{p})$$

$\boldsymbol{\Psi}$ should be of class C^k, $k \geq n$, n being the relative degree of the robot's state model.

It is given in the $\mathcal{R}_t$ frame.

Problem formalisation: state model

Let us consider that the following variables are known:

- $\mathbf{x}_r = (\mathbf{p}_r, \mathbf{v}_{r,\mathcal{R}_r}, \boldsymbol{\xi})^{\mathrm{T}}$, state of the robot
- $\mathbf{p}_t$, $\mathbf{v}_t$, $\mathbf{a}_t$, $\boldsymbol{\xi}$, $\boldsymbol{\omega}_t$, $\dot{\boldsymbol{\omega}}_t$

And that we have a state model for the robot:

$$\dot{\mathbf{x}}_r = \mathbf{f}\left(\mathbf{x}_r, \mathbf{u}_r\right)$$

The goal is to find $\mathbf{u}_r$ so that the robot follows $\boldsymbol{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right)$: the output vector $\mathbf{y}$ must be driven to $\mathbf{0}$.

$$\mathbf{y} = \begin{pmatrix} \mathbf{v}_r - \boldsymbol{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right)_{\mathcal{R}_r} \\ \xi - \boldsymbol{\Xi}\left(\boldsymbol{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right)_{\mathcal{R}_0}\right) \end{pmatrix}$$

Problem formalisation: Vector field transformation

The vector field $\mathbf{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right)$ is given in the frame $\mathcal{R}_t$, but must be expressed in $\mathcal{R}_r$ to be used to drive the robot:

$$\mathbf{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right)_{\mathcal{R}_r} = \mathbf{R}_r^{\mathrm{T}} \cdot \mathbf{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right)_{\mathcal{R}_0}$$
$$\mathbf{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right)_{\mathcal{R}_0} = \mathbf{R}_t \cdot \mathbf{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right) + \mathbf{R}_t \cdot \mathbf{v}_t + \mathbf{Ad}\left(\omega_t\right) \cdot \left(\mathbf{p}_r - \mathbf{p}_t\right)$$

Problem formalisation: state-feedback linearisation

Using a state-feedback linearisation method ([Jaulin, 2015]), the command vector $\mathbf{u}_r$ can be computed as follows:

$$\mathbf{u}_r = \mathbf{A}^{-1}(\mathbf{x}_r) \cdot \left(\mathcal{E}\left(\mathbf{y} \dots \mathbf{y}^{(n-1)}\right) - \mathbf{b}(\mathbf{x}_r)\right)$$

where $\mathbf{y}^{(n)} = \mathcal{E}\left(\mathbf{y} \dots \mathbf{y}^{(n-1)}\right)$ is the chosen error dynamics equation.

State model: 2D holonomic robot

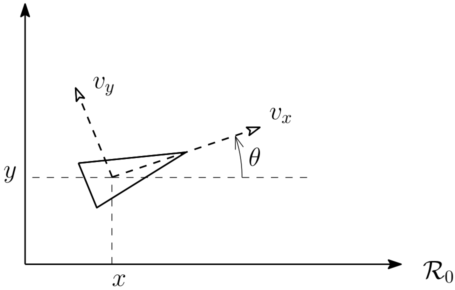


Figure: State model of the robot

Let us consider the following state model for our holonomic 2D robot:

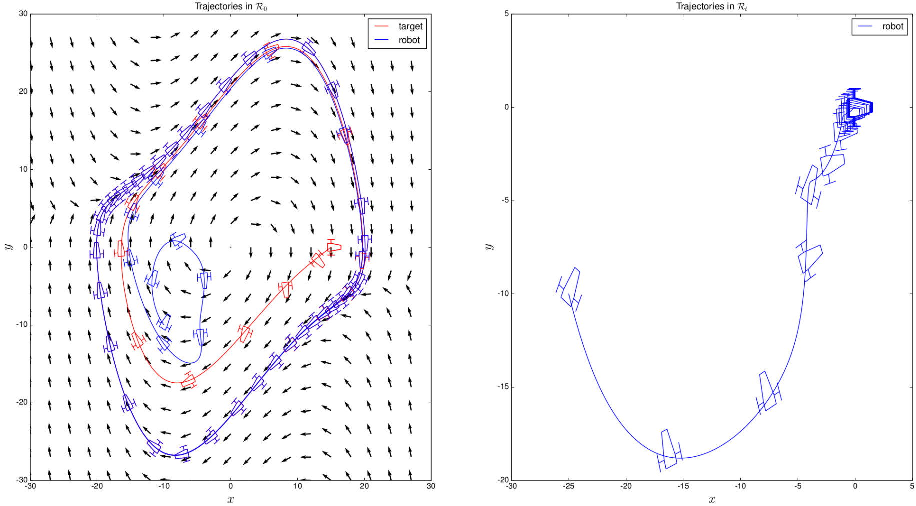
$$\mathbf{x} = (x, y, v_x, v_y, \theta)^{\mathrm{T}}$$

$$\dot{\mathbf{x}} = \begin{pmatrix} \mathbf{R}(\theta) \cdot \mathbf{v} \\ \mathbf{a} - \mathbf{Ad}(\omega) \cdot \mathbf{v} \\ \omega \end{pmatrix}$$

$$\mathbf{u} = (a_x, a_y, \omega)^{\mathrm{T}}$$

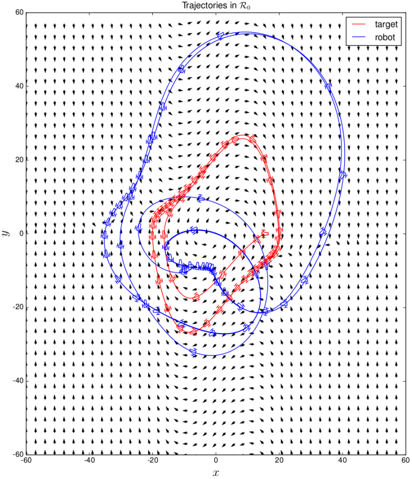
Simple attractive field

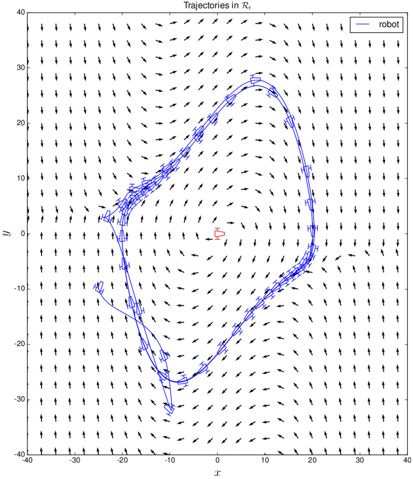
$$\boldsymbol{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right)_{\mathcal{R}_t} = -\mathbf{p}_{r,\mathcal{R}_t}$$



Van der Pol cycle

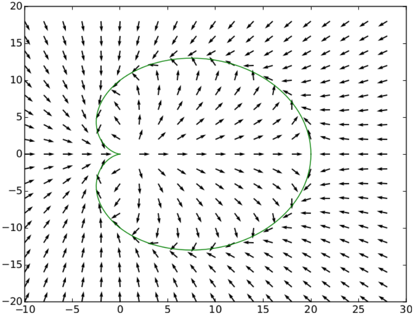
$$\mathbf{\Psi}(x,y)_{\mathcal{R}_t} = \begin{pmatrix} y \\ -(0.01x^2-1)\,y-x \end{pmatrix}$$



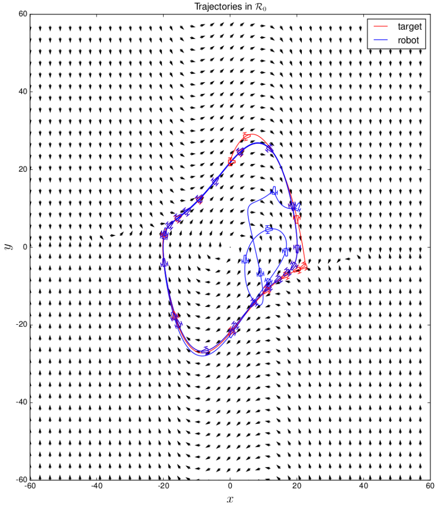


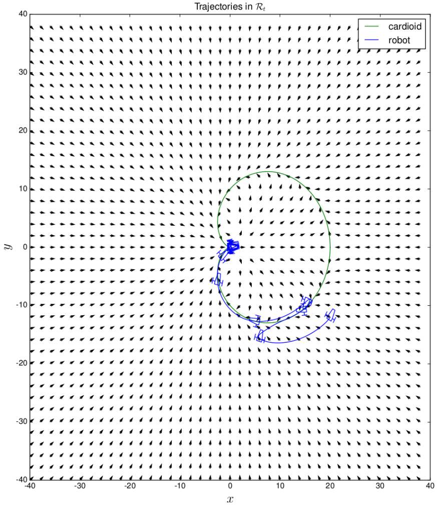
Cardioid vector field I

$$\theta = \arctan2\left(\mathbf{p}_{r,y}, \mathbf{p}_{r,x}\right)$$
$$t_x\left(\theta\right) = -R\left(\sin\left(2\theta\right) + \sin\left(\theta\right)\right)$$
$$t_y\left(\theta\right) = R\left(\cos\left(2\theta\right) + \cos\left(\theta\right)\right)$$
$$r\left(\theta\right) = R\left(1 + \cos\left(\theta\right)\right)$$
$$\mathbf{\Psi}\left(\mathbf{p}_{r,\mathcal{R}_t}\right)_{\mathcal{R}_t} = \mathrm{sign}\left(\theta\right)\begin{pmatrix} t_x \\ t_y \end{pmatrix} - \left(\|\mathbf{p}_r\| - r\left(\theta\right)\right)\mathbf{p}_r$$



Cardioid vector field II





Conclusion

Conclusion

- Promising method for docking problems :
  - Improved robustness w.r.t. environment's disturbances
  - No overshoot phenomenon
  - Anticipates the target's moves
  - Mathematically simple to derive and implement
  - Drives the robot along a given vector field, which can be tuned to suit every need
- Limitations :
  - Requires generally a vector field of class $\mathcal{C}^2$, sometimes more
  - A state model of the robot is required
  - The position/orientation, linear/angular velocities and accelerations of the target must be known
  - Finding a suitable vector field can be a bit tricky

Conclusion

- Promising method for docking problems :
  - Improved robustness w.r.t. environment's disturbances
  - No overshoot phenomenon
  - Anticipates the target's moves
  - Mathematically simple to derive and implement
  - Drives the robot along a given vector field, which can be tuned to suit every need
- Limitations :
  - Requires generally a vector field of class C^2, sometimes more
  - A state model of the robot is required
  - The position/orientation, linear/angular velocities and accelerations of the target must be known
  - Finding a suitable vector field can be a bit tricky

Conclusion

- Promising method for docking problems :
  - Improved robustness w.r.t. environment's disturbances
  - No overshoot phenomenon
  - Anticipates the target's moves
  - Mathematically simple to derive and implement
  - Drives the robot along a given vector field, which can be tuned to suit every need
- Limitations :
  - Requires generally a vector field of class C^2, sometimes more
  - A state model of the robot is required
  - The position/orientation, linear/angular velocities and accelerations of the target must be known
  - Finding a suitable vector field can be a bit tricky

Conclusion

- Promising method for docking problems :
  - Improved robustness w.r.t. environment's disturbances
  - No overshoot phenomenon
  - Anticipates the target's moves
  - Mathematically simple to derive and implement
  - Drives the robot along a given vector field, which can be tuned to suit every need
- Limitations :
  - Requires generally a vector field of class $\mathcal{C}^2$, sometimes more
  - A state model of the robot is required
  - The position/orientation, linear/angular velocities and accelerations of the target must be known
  - Finding a suitable vector field can be a bit tricky

Conclusion

- Promising method for docking problems :
  - Improved robustness w.r.t. environment's disturbances
  - No overshoot phenomenon
  - Anticipates the target's moves
  - Mathematically simple to derive and implement
  - Drives the robot along a given vector field, which can be tuned to suit every need
- Limitations :
  - Requires generally a vector field of class $\mathcal{C}^2$, sometimes more
  - A state model of the robot is required
  - The position/orientation, linear/angular velocities and accelerations of the target must be known
  - Finding a suitable vector field can be a bit tricky

Conclusion

- Promising method for docking problems :
  - Improved robustness w.r.t. environment's disturbances
  - No overshoot phenomenon
  - Anticipates the target's moves
  - Mathematically simple to derive and implement
  - Drives the robot along a given vector field, which can be tuned to suit every need
- Limitations :
  - Requires generally a vector field of class $\mathcal{C}^2$, sometimes more
  - A state model of the robot is required
  - The position/orientation, linear/angular velocities and accelerations of the target must be known
  - Finding a suitable vector field can be a bit tricky

Conclusion

- Promising method for docking problems :
  - Improved robustness w.r.t. environment's disturbances
  - No overshoot phenomenon
  - Anticipates the target's moves
  - Mathematically simple to derive and implement
  - Drives the robot along a given vector field, which can be tuned to suit every need
- Limitations :
  - Requires generally a vector field of class $\mathcal{C}^2$, sometimes more
  - A state model of the robot is required
  - The position/orientation, linear/angular velocities and accelerations of the target must be known
  - Finding a suitable vector field can be a bit tricky

Conclusion

- Promising method for docking problems :
  - Improved robustness w.r.t. environment's disturbances
  - No overshoot phenomenon
  - Anticipates the target's moves
  - Mathematically simple to derive and implement
  - Drives the robot along a given vector field, which can be tuned to suit every need
- Limitations :
  - Requires generally a vector field of class $\mathcal{C}^2$, sometimes more
  - A state model of the robot is required
  - The position/orientation, linear/angular velocities and accelerations of the target must be known
  - Finding a suitable vector field can be a bit tricky

Conclusion

- Promising method for docking problems :
  - Improved robustness w.r.t. environment's disturbances
  - No overshoot phenomenon
  - Anticipates the target's moves
  - Mathematically simple to derive and implement
  - Drives the robot along a given vector field, which can be tuned to suit every need
- Limitations :
  - Requires generally a vector field of class $\mathcal{C}^2$, sometimes more
  - A state model of the robot is required
  - The position/orientation, linear/angular velocities and accelerations of the target must be known
  - Finding a suitable vector field can be a bit tricky

Future research

- Find an elegant expression for a docking vector field
- Develop a method based on Interval Analysis to validate the vector field w.r.t. hardware limitations

Future research

- Find an elegant expression for a docking vector field
- Develop a method based on Interval Analysis to validate the vector field w.r.t. hardware limitations

References

Luc Jaulin (2015)
Mobile robotics
ISTE WILEY

Maël Le Gallic, Joris Tillet, Luc Jaulin and Fabrice Le Bars (2018)
Tight slalom control for sailboat robots