Université Mohammed V
Faculté des Sciences
Rabat

Introduction à l'algorithme de Braitenberg

Pr. Younès Raoui

01 Origine de l'algorithme de Braitenberg

Développement historique de l'algorithme

Théorie de Braitenberg et ses premières applications

L'algorithme de Braitenberg trouve son origine dans les travaux pionniers de Valentino Braitenberg en cybernétique et en robotique. Ses premières applications ont été orientées vers la compréhension des comportements autonomes.

Influence de la cybernétique sur l'algorithme

L'algorithme de Braitenberg a été fortement influencé par les concepts de la cybernétique, notamment en ce qui concerne la relation entre les agents autonomes et leur environnement.

Évolution des principes fondamentaux

Au fil du temps, l'algorithme de Braitenberg a évolué pour inclure des principes fondamentaux de perception et de réactivité chez les agents artificiels.

Applications contemporaines de l'algorithme

Utilisation dans la robotique moderne

L'algorithme de Braitenberg est actuellement utilisé dans le développement de robots autonomes capables d'interagir avec leur environnement de manière sophistiquée.

Intégration dans les systèmes d'intelligence artificielle

Il a été intégré dans les systèmes d'intelligence artificielle pour améliorer la compréhension et la modélisation des comportements complexes.

Influence sur la recherche en neurosciences computationnelles

L'algorithme de Braitenberg a eu un impact significatif sur la recherche en neurosciences computationnelles en fournissant des modèles informatiques pour étudier les interactions cerveau-environnement.

Implications philosophiques de l'algorithme

Réflexions sur l'autonomie et l'interaction

L'algorithme de Braitenberg suscite des réflexions profondes sur la nature de l'autonomie et de l'interaction chez les systèmes artificiels et naturels.

Exploration des frontières entre l'homme et la machine

Il offre un cadre pour explorer les frontières entre l'homme et la machine, remettant en question les conceptions traditionnelles de la cognition et de l'intelligence.

Impact sur la perception de l'agentivité

L'algorithme de Braitenberg a des implications profondes sur la perception de l'agentivité et de l'auto-détermination dans un contexte technologique en évolution constante.

02 Principes de l'algorithme de Braitenberg

Exploration des comportements basés sur l'entrée de l'utilisateur

Réactivité sensorielle

La réactivité sensorielle est un aspect clé de l'algorithme de Braitenberg, où les entrées sensorielles influencent directement les comportements des agents.

Interaction avec l'environnement

Les agents réagissent à leur environnement en fonction des entrées sensorielles, ce qui influence leur mouvement et leurs actions.

Adaptation aux stimuli

Les agents peuvent s'adapter et réagir différemment aux stimuli sensoriels en fonction de leurs caractéristiques et de leur expérience préalable.

Modélisation des réponses des agents aux entrées de l'utilisateur

Réponses non linéaires

Les réponses des agents peuvent être non linéaires, ce qui signifie que les comportements ne sont pas simplement proportionnels aux entrées sensorielles.

Effets de seuil

Certains agents peuvent présenter des effets de seuil, où seuls les stimuli dépassant un certain seuil déclenchent une réponse significative.

Influence des connexions neuronales

Les connexions neuronales des agents jouent un rôle crucial dans la façon dont ils interprètent et réagissent aux entrées sensorielles.

Applications pratiques de l'algorithme de Braitenberg

Robotique autonome

L'algorithme de Braitenberg a des applications importantes en robotique autonome, où les robots peuvent réagir de manière adaptative à leur environnement.

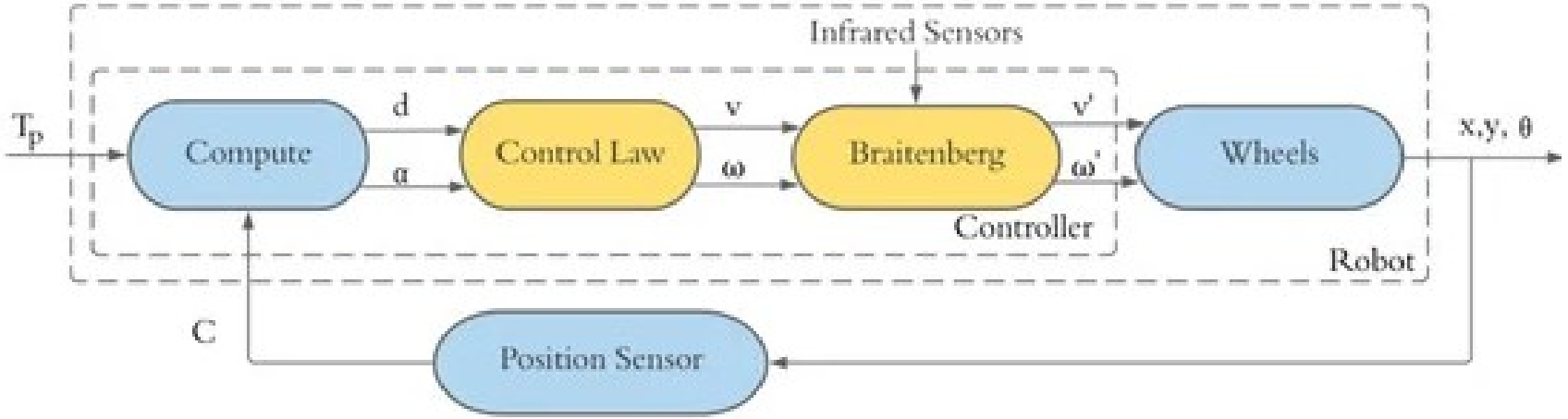
Exploration de l'intelligence artificielle

Les principes de l'algorithme de Braitenberg sont étudiés dans le domaine de l'intelligence artificielle pour comprendre les comportements réactifs.

Études comportementales

Les modèles basés sur l'algorithme de Braitenberg sont utilisés pour étudier et comprendre les comportements réactifs dans des contextes divers.

Formalisme de navigation



Position Sensor: This block measures the current position (x, y, θ) and feeds it back to the Compute block for further processing.

Mathematical Representation

Compute Block:

The target position T_p is used to compute the distance d and angle α.

$$d = \sqrt{(x_t - x)^2 + (y_t - y)^2}$$
$$\alpha = \arctan\left(\frac{y_t - y}{x_t - x}\right) - \theta$$

where (x_t, y_t) is the target position and (x, y) is the current position.

Control Law Block:

The control law computes the linear velocity v and angular velocity ω based on d and α.

$$v = k_d \cdot d$$
$$\omega = k_\alpha \cdot \alpha$$

where k_d and k_α are control gains.

Braitenberg Controller:

Adjusts v and ω based on sensor inputs to v' and ω'.

$$v' = f(v, \text{sensor inputs})$$
$$\omega' = g(\omega, \text{sensor inputs})$$

Wheels Block:

The robot's motion is described by the kinematic equations.

$$\dot{x} = v' \cos(\theta)$$
$$\dot{y} = v' \sin(\theta)$$
$$\dot{\theta} = \omega'$$

Dynamic Neural Fields (DNFs)

Dynamic Neural Fields (DNFs) are mathematical models used to describe the evolution of neural activation over time, typically in the context of spatially or topologically organized neural populations. The basic equation of a dynamic neural field can be represented as follows:

$$\frac{\partial u(x,t)}{\partial t} = -u(x,t) + \int_{\Omega} w(x,x')S(u(x',t))dx' + I(x,t)$$

where:

$u(x,t)$ is the activation level of the field at position x and time t,
Ω is the domain of the neural field,
$w(x,x')$ is the synaptic weight function describing the connection strength between neurons at positions x and x',
$S(u)$ is a nonlinear activation function, often a sigmoid function, that transforms the field's activation level into a firing rate,
$I(x,t)$ is the external input to the field at position x and time t.

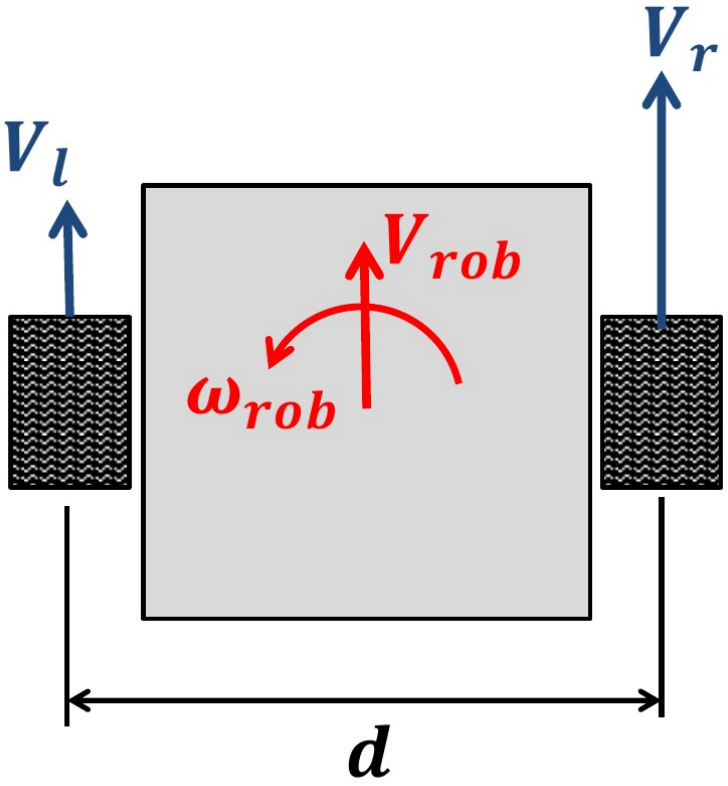
Représentation : Le champ neuronal représente l'environnement spatial autour de l'agent, les niveaux d'activation indiquant l'opportunité de se déplacer dans différentes directions.

Intégration : Grâce au terme intégral, le champ intègre l'information dans l'espace, ce qui permet une représentation lisse et continue de l'environnement qui prend en compte les effets des obstacles multiples.

Dynamique : L'évolution temporelle du champ permet une mise à jour dynamique de la représentation au fur et à mesure que l'agent se déplace et que des obstacles apparaissent ou se déplacent.

Prise de décision : L'agent utilise le gradient du champ neuronal pour décider où se déplacer, évitant ainsi les obstacles en choisissant des directions plus désirables.
Ainsi, les champs neuronaux dynamiques fournissent un cadre pour l'intégration d'informations spatiales dans le temps et la prise de décisions continues concernant le mouvement, ce qui les rend particulièrement utiles pour des tâches telles que l'évitement d'obstacles dans les robots mobiles ou les véhicules autonomes.

Differential Drive Kinematics



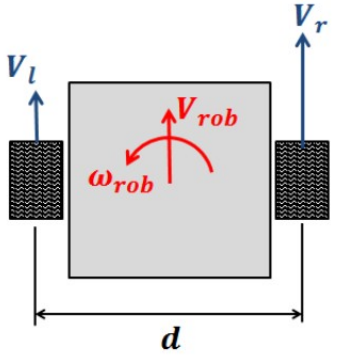
Differential Drive Velocity

- Velocity:

$$V_{rob} = \frac{V_r + V_l}{2}$$

- Rotational Speed:

$$\omega_{rob} = \frac{V_r - V_l}{d}$$

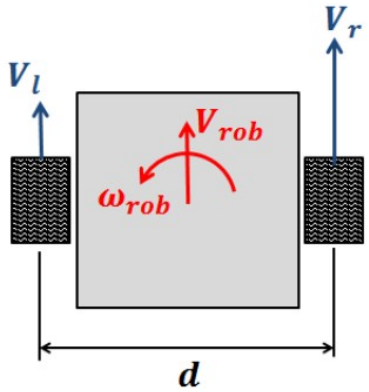


Differential Drive Velocity

- Solve for V_r, V_l:

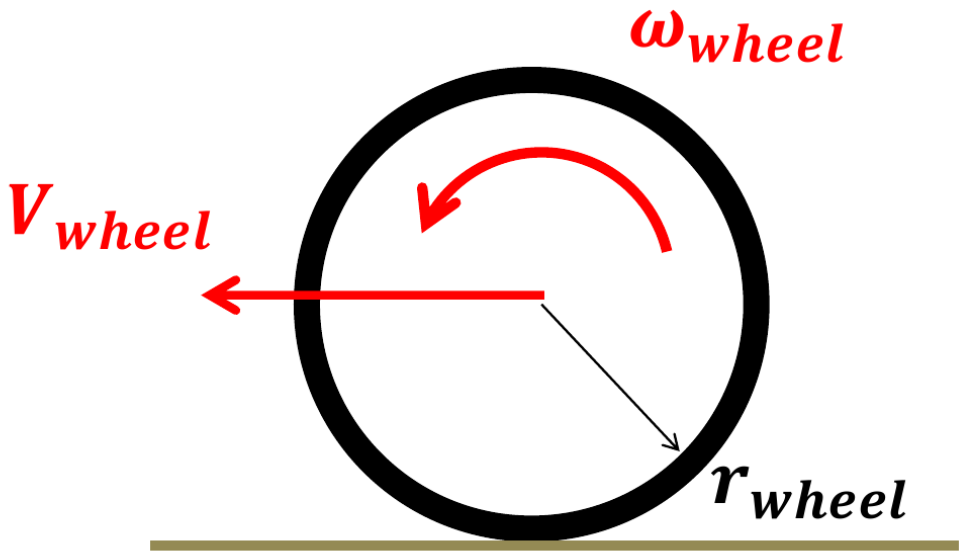
$$V_r = V_{rob} + \frac{d}{2}\omega_{rob}$$

$$V_l = V_{rob} - \frac{d}{2}\omega_{rob}$$



Wheel Rotation Velocity Relationship:

$$V_{wheel} = \omega_{wheel} r_{wheel}$$



- Assume no slipping

Desired velocity: v_des

Desired Rotation rate: om_des

```
--Lua Code
v_des=0.1
om_des=0.1
d=0.06                --wheels separation

v_r=(v_des+d*om_des)
v_l=(v_des-d*om_des)
```

Desired velocity: v_des

Desired Rotation rate: om_des

```
--Lua Code


r_w=0.0275        --wheel radius
omega_right=v_r/r_w
omega_left=v_l/r_w
```

Thank You