véhicule autonome avec lidar 2D

Clemens Valens (Elektor)

Avec un lidar qui détecte les obstacles environnants à distance, un robot (ou un véhicule autonome) jouit alors d'une liberté de mouvement bien plus grande. Pour me familiariser avec cette technique, j'ai construit un simple chariot télécommandé et l'ai laissé se déplacer dans mon salon.

Lidar est l'acronyme de ***li**ght **d**etection **a**nd **r**anging* (détection et télémétrie par la lumière). Le lidar est un radar infrarouge (IR), c.-à-d. travaillant dans la partie optique du spectre électromagnétique. La source IR est un laser. Un lidar émet des impulsions IR et mesure le temps qu'elles mettent à revenir en cas de réflexion par un objet distant. La vitesse de la lumière étant une constante connue, la distance de l'objet peut être calculée à partir du temps aller-retour des impulsions (**fig. 1**).

Lidar bidimensionnel

Un lidar peut être unidimensionnel (1D), comme un télémètre laser. Il peut aussi être bidimensionnel (2D), comme un radar de navire ou de tour de contrôle du trafic aérien. Enfin, le lidar peut être tridimensionnel (3D), par ex. sur un avion pour calculer le rendu en relief de la surface de la terre survolée. Ce projet utilise un lidar 2D.

Fondamentalement, un lidar 2D est un lidar 1D rotatif. Au lieu de faire tourner l'ensemble laser/détecteur, il est souvent plus facile de diriger les IR vers un miroir rotatif. Avec des impulsions IR périodiques, le lidar couvre 360° et crée une carte des distances centrée sur lui. La réflectivité des objets est fondamentale. Le corps noir idéal, qui ne réfléchit rien, ne pourrait pas être vu par un lidar.

Connecter le lidar

Pour mon expérimentation, j'ai utilisé le X4 de Ydlidar (**fig. 2**). Il a une portée atteignant 10 m et une résolution angulaire de 0,5° (pour une distance de 50 cm max.). Il embarque un laser infrarouge à 785 nm de longueur d'onde. Il n'utilise pas de miroir rotatif, mais fait tourner l'ensemble laser/détecteur.

Le lidar X4 sort les distances en continu sur un port série à 128 000 bauds. Il est également piloté via cette liaison série. Des commandes simples démarrent et arrêtent le balayage. Il envoie des informations sur demande. Deux fils auxiliaires gèrent séparément les fonctions marche/arrêt et vitesse du moteur. Donc, pour

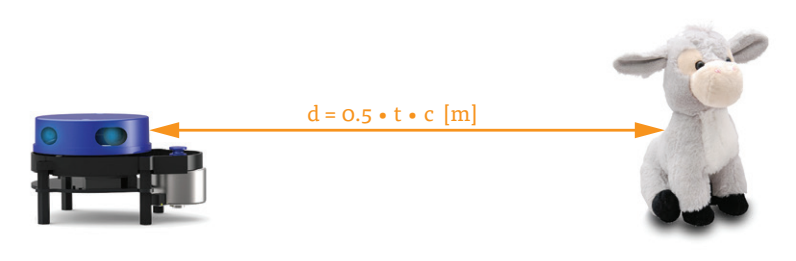


Figure 1. Formule de calcul de la distance. t = temps de parcours de l'impulsion lumineuse en s et c = vitesse de la lumière en m/s. Le facteur 0,5 tient compte de l'aller-retour jusqu'à l'objet c.-à-d. du fait que la distance est parcourue deux fois.

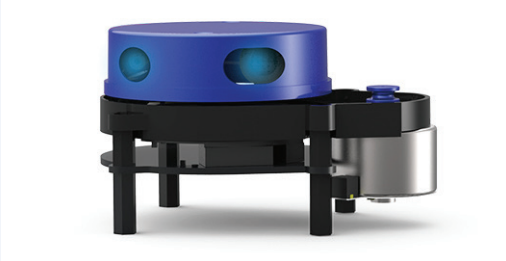


Figure 2. Le YDLIDAR X4 est un lidar 2D peu coûteux dont la portée et la précision sont suffisantes pour qu'un robot se déplace sans heurter d'objets.

que ce lidar fonctionne en mode 1D, il suffit d'arrêter le moteur.
J'ai connecté le port série et les connexions du moteur à un kit ESP32 Pico (**fig. 3**). Pour économiser l'énergie, je n'ai utilisé que la vitesse la plus lente du moteur, soit 400 tr/min. Le lidar consomme alors environ 400 mA.

Analyse des données du lidar

J'ai programmé le kit ESP32 Pico sous EDI Arduino. J'ai écrit une fonction d'analyse des données du lidar, puis vérifié leur interprétation, car le manuel de développement du X4 n'est pas très clair sur le processus et spécifie deux niveaux de détail. Le second niveau offre une meilleure résolution angulaire, mais implique beaucoup de calculs de tangentes inverses, qui accaparent le CPU. J'ai donc d'abord essayé l'interprétation simple.
J'ai placé le lidar sur une table dans un espace rectangulaire clos et je l'ai laissé tourner un moment. Une fois arrêté, je lui ai fait envoyer en valeurs séparées par des virgules (CSV), via un port série, un balayage de 360° moyen que j'ai chargé dans Excel. J'ai pu l'afficher (**fig. 4**) avec la fonction graphique radar. Le résultat fut correct tant en forme (un rectangle) qu'en respect des distances, je n'ai donc pas tenté de l'améliorer en ajoutant des calculs de tangente inverse.

Construire un petit robot

Pour l'étape suivante, j'ai construit un simple chariot télécommandé sur lequel j'ai monté le lidar. Le chariot a deux roues motorisées au centre et une béquille à chaque extrémité.

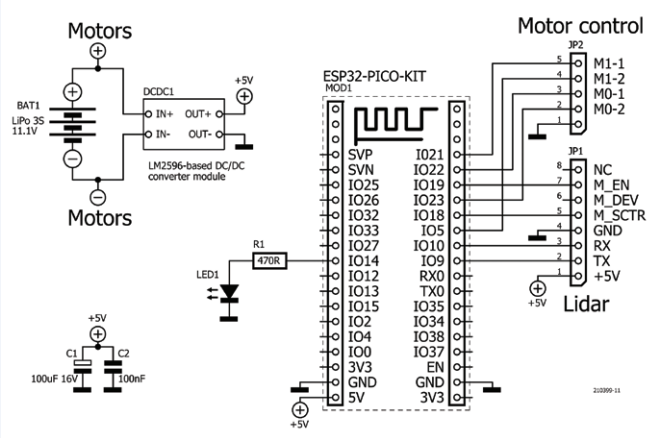


Figure 3. Le kit ESP32 Pico est utilisé comme cerveau du véhicule autonome. La commande des moteurs et le convertisseur CC-CC sont des modules génériques en vente en ligne.

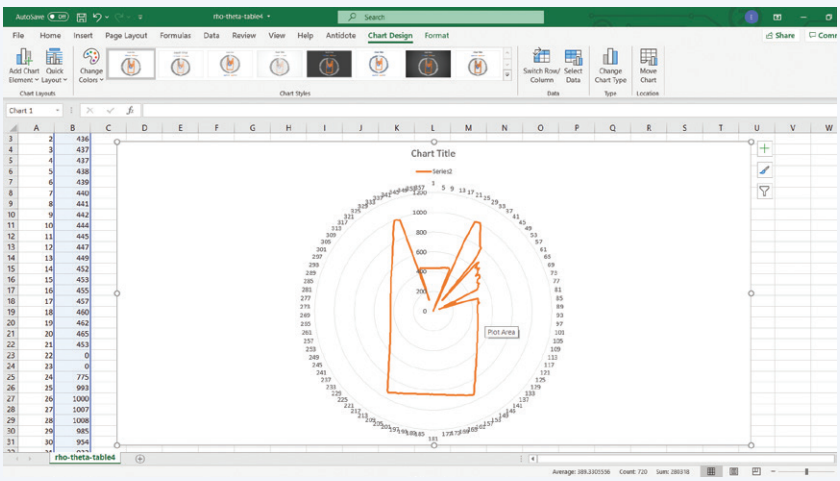


Figure 4. J'ai pu vérifier rapidement mon interprétation des données lidar avec Excel de Microsoft. La courte ligne horizontale à 400 mm (et l'ombre derrière elle) vient de mon ordinateur portable qui enregistre les données du lidar décodées.

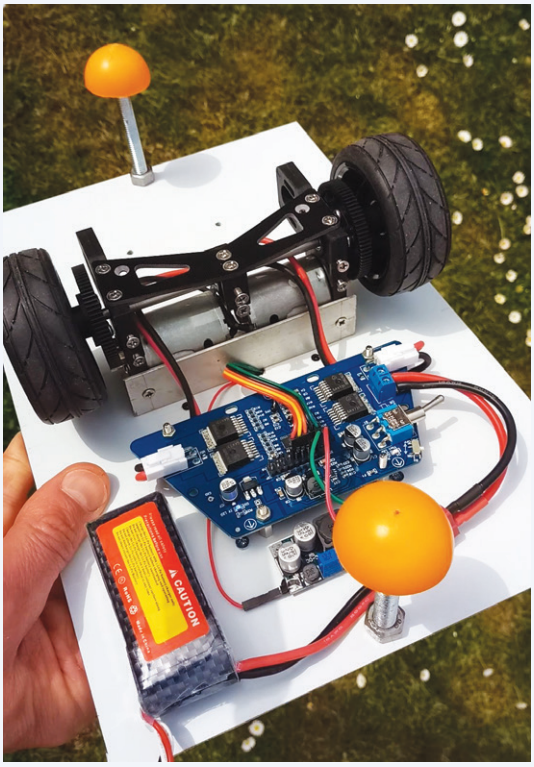
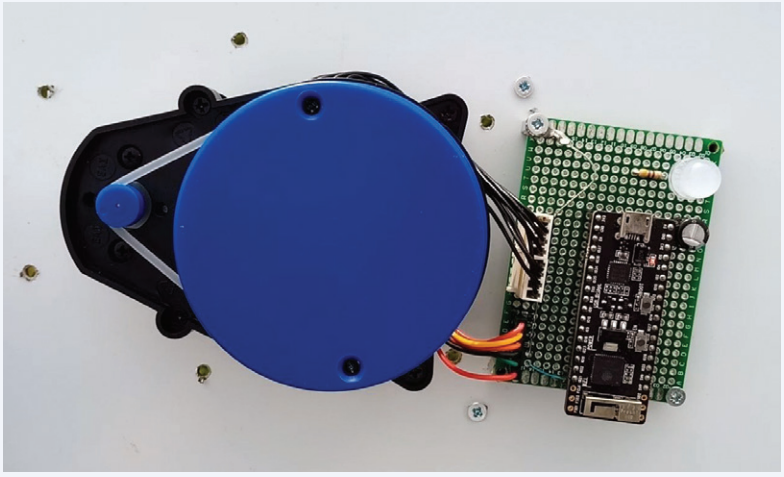


Figure 5. Vue du dessous du chariot. L'ensemble moteur et sa carte de commande viennent de chez Landzo.com mais ne sont plus disponibles.

Figure 6. Face supérieure du véhicule montrant le kit ESP32 Pico et le lidar.



J'ai collé une demi-balle de ping-pong aux béquilles pour réduire les frottements. Tout est monté sur une plaque pour circuit imprimé cuivre double face FR4. La face inférieure accueille les roues avec les moteurs, leur carte de commande et l'alimentation, une batterie Li-Po 3S 11,1 V (**fig. 5**). La face supérieure accueille le module ESP32 et le lidar **(fig. 6**). La plaque est mise à la terre, ce qui protège l'ESP32 des parasites des moteurs. Le centre du lidar et celui de la plaque de montage coïncident avec l'axe vertical du train de roues. Ce chariot simple peut tourner sur lui-même et est assez agile et maniable.

Ajouter une télécommande via Bluetooth

La bibliothèque gratuite et *open source* Dabble [1] fournit le contrôle Bluetooth de l'ESP32 et de l'Arduino, ainsi qu'une application de télécommande pour smartphone avec diverses surfaces de contrôle (**fig. 7)**. L'une de ces surfaces est une manette de jeu, c'était idéal pour mon application. Elle est très facile à utiliser et j'ai pu piloter le chariot avec mon téléphone.

Un algorithme de recherche de chemin

Mon but était de programmer le chariot pour qu'il se déplace tout seul, sans heurter aucun objet, par ex. des meubles. Une approche prisée consiste à laisser le chariot se déplacer et reculer ou s'éloigner dès qu'il s'approche trop près d'un objet, mais cela exige qu'il prenne des décisions. Je voulais quelque chose de plus simple. Beaucoup d'algorithmes simples conduisent à un comportement complexe, par ex. celui d'une nuée d'oiseaux volant ensemble [2]. Je voulais quelque chose de ce genre.

Mon idée fut de faire en sorte que le chariot aille toujours dans la direction de la plus grande distance donnée par le lidar. Pour éviter de tourner en rond, il ne regarde que vers l'avant, dans la plage de −90° à +90°. Il fut assez facile de mettre en œuvre cette règle. À chaque balayage, on met à jour une table avec la distance moyenne à chaque degré. La table comporte donc 360 entrées, une par degré. Dans cette table, on recherche ensuite l'arc de 10° qui donne la plus grande distance moyenne (le choix de 10° relève d'une décision plutôt arbitraire). Le degré médian de cet arc est la direction que le chariot doit prendre. Pour cela, le chariot tourne jusqu'à ce que le degré médian, la bonne direction, tombe à zéro. Il s'agit donc d'un algorithme de commande classique qui tente de minimiser une « erreur » (**fig. 8**).

Premier essai

À ma grande surprise, avec cet algorithme simple, le chariot a réussi du premier coup à se déplacer dans notre salon sans rien heurter (**fig. 9)**. Il a fait le tour du canapé et traversé des passages étroits sans difficulté. Le chariot ne connaît rien de son environnement ni de lui-même, par ex. ses dimensions. De plus, je n'ai tenté aucune optimisation. J'ai juste pris les valeurs que j'ai jugées raisonnables pour chaque paramètre (vitesse de translation, de virage et angle de recherche).

La télécommande Bluetooth est très pratique pour induire le chariot en erreur ou l'aider à se sortir de situations délicates. Elle peut aussi servir à ajuster les paramètres à la volée.

Comme je suis bien davantage motivé par les preuves de concept que par l'optimisation,

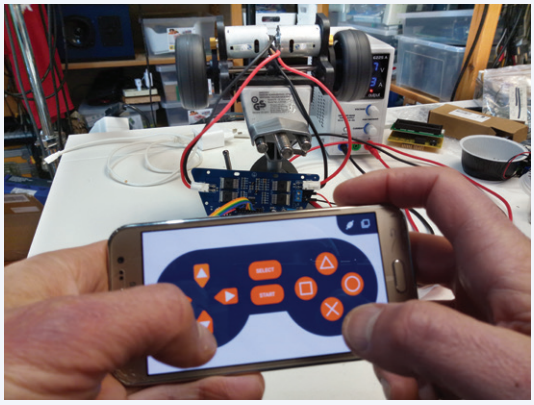


Figure 7. Essai sur établi de la télécommande pour smartphone basée sur Dabble.

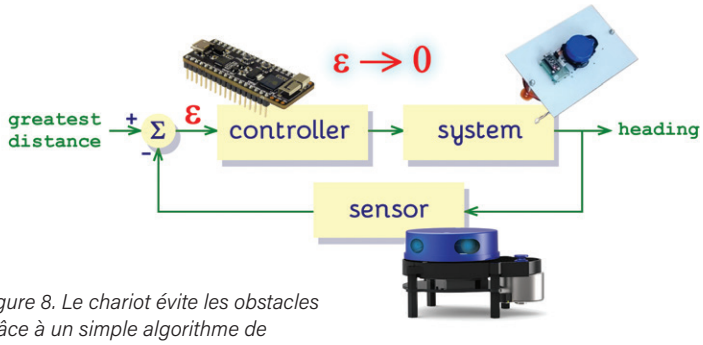


Figure 8. Le chariot évite les obstacles grâce à un simple algorithme de recherche d'un écart minimal.

Figure 9. C'est parti ! Regardez la vidéo [4] pour apprécier la fluidité de déplacement.

je me suis arrêté à ce stade. Si la mise au point vous tente, vous trouverez ci-après les liens vers le code. Les possibilités d'amélioration de ce prototype (très loin d'un aspirateur ou d'une tondeuse à gazon autonome) sont nombreuses, mais les résultats obtenus sont très encourageants. Le logiciel de ce projet, un croquis Arduino pour l'ESP32 [3] est téléchargeable. Une vidéo [4] est disponible ainsi qu'un bonus [5].

210399-04

Contributeurs

Idée, conception, texte et photographies : **Clemens Valens**
Rédaction : **Jens Nickel, C. J. Abate**
Mise en page : **Giel Dols**
Traduction : **Yves Georges**

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (clemens.valens@elektor.com) ou contactez Elektor (redaction@elektor.fr).

PRODUITS

- **Kit ESP32 Pico**
  www.elektor.fr/18423
- **YDLIDAR X4**
  www.elektor.fr/18601

LIENS

[1] Dabble : https://thestempedia.com/product/dabble/
[2] Comportement grégaire : https://fr.wikipedia.org/wiki/Comportement_gr%C3%A9gaire
[3] Croquis Arduino ESP32 pour ce projet : https://github.com/ClemensAtElektor/Lidar-controlled-autonomous-vehicle/
[4] Ce projet en vidéo : https://youtu.be/BmNelv_gR9Q
[5] Feux de stationnement basés sur un lidar par Rob Reynolds de SparkFun : https://youtu.be/KRfidalgJx8