Avec l'aide de l'intelligence artificielle

TROISIÈME PARTIE : LE REPÉRAGE DES TRAINS

EX MACHINA

Un réseau automatisé

Après avoir construit le tableau de commande de notre locodrome, nous allons commencer à penser sécurité, avec la mise en place du système empêchant le rattrapage de deux trains.

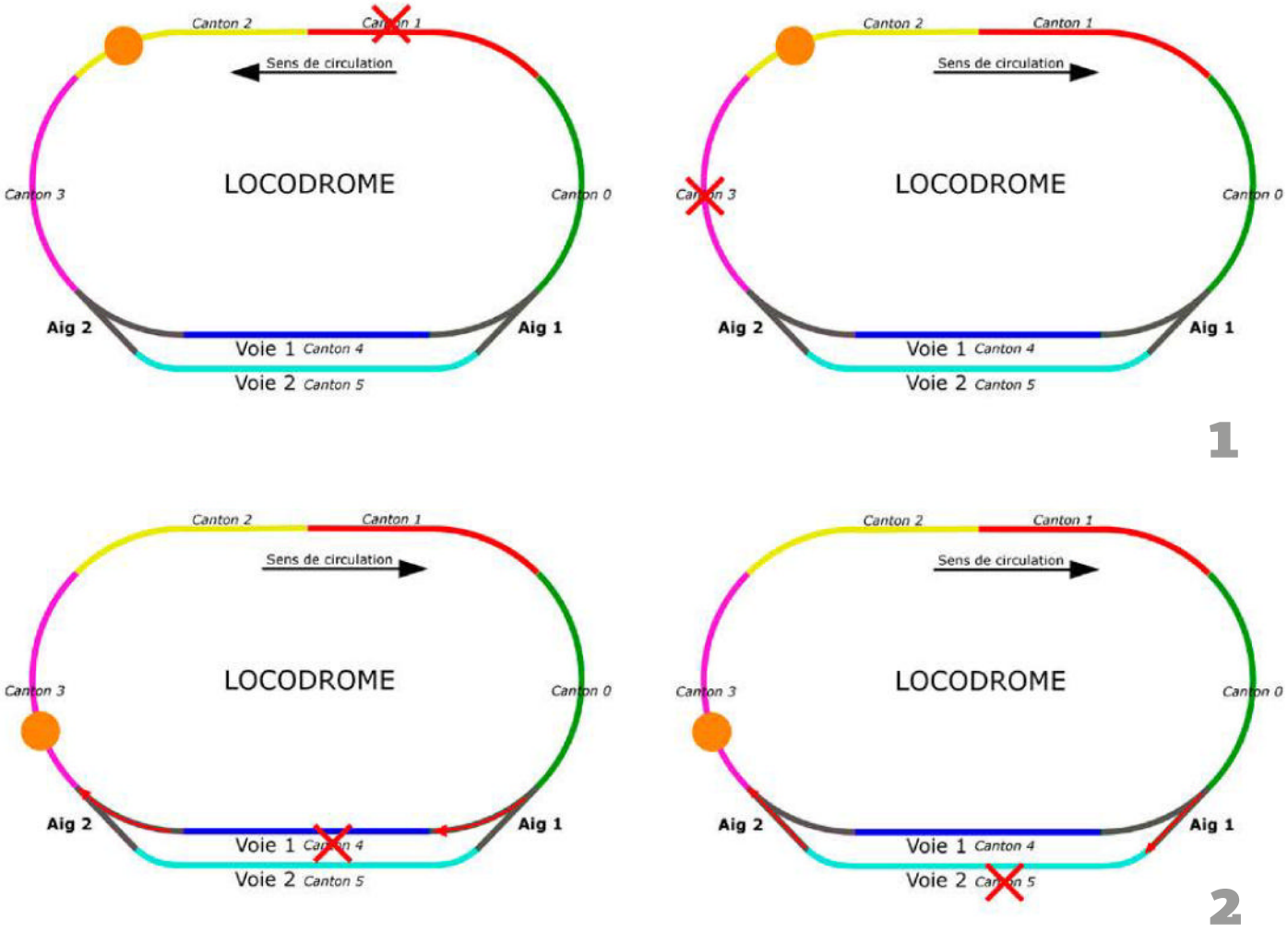
Texte et illustrations: Christian Bézanger

Dans cet article, nous allons voir comment repérer nos trains au fur et à mesure de leur progression. Nous expliquerons le principe du B.A.L et comment le transposer sur un réseau analogique. Et nous serons capables, à la fin de cet article de lancer deux trains pour qu'ils se suivent sans jamais se rattraper.

Principe du Block Automatique Lumineux (B.A.L)

En analogique, le principe du B.A.L est simple : lorsqu'un canton est occupé, on coupe le courant sur la zone d'arrêt du canton précédent et on met le signal lumineux au rouge (sémaphore). La locomotive s'arrête devant le signal qui doit être judicieusement placé.

Ce signal, qui a trois feux, est vert (voie libre) si les deux cantons qui suivent sont libres, et orange (avertissement) si le canton suivant est libre mais que le canton d'après est occupé. En modélisme, un canton est donc une portion de voie constitué d'une zone de pleine voie (ZPV sur laquelle le train peut rouler librement) et d'une zone d'arrêt (ZA alimentée ou non en fonction du signal). Le train s'arrête brutalement sur la zone d'arrêt et avec l'arrivée de l'électronique, on a ajouté une zone de ralentissement permettant au train de réduire sa vitesse si le signal est orange ou rouge. Il faut des cantons très longs pour disposer des trois zones et ce n'est pas le cas de ce locodrome. Aussi bizarre que cela paraisse, nos cantons ne seront constitués que de la zone d'arrêt. Donc, si un canton



est occupé, il faut couper l'alimentation en courant du canton précédent, rôle dévolu à notre carte 8 relais (voir l'article 2).

Mais qu'appelle-t-on canton précédent ? La **figure 1** montre que ce n'est pas si évident que cela car notre voie peut être parcourue dans les deux sens.

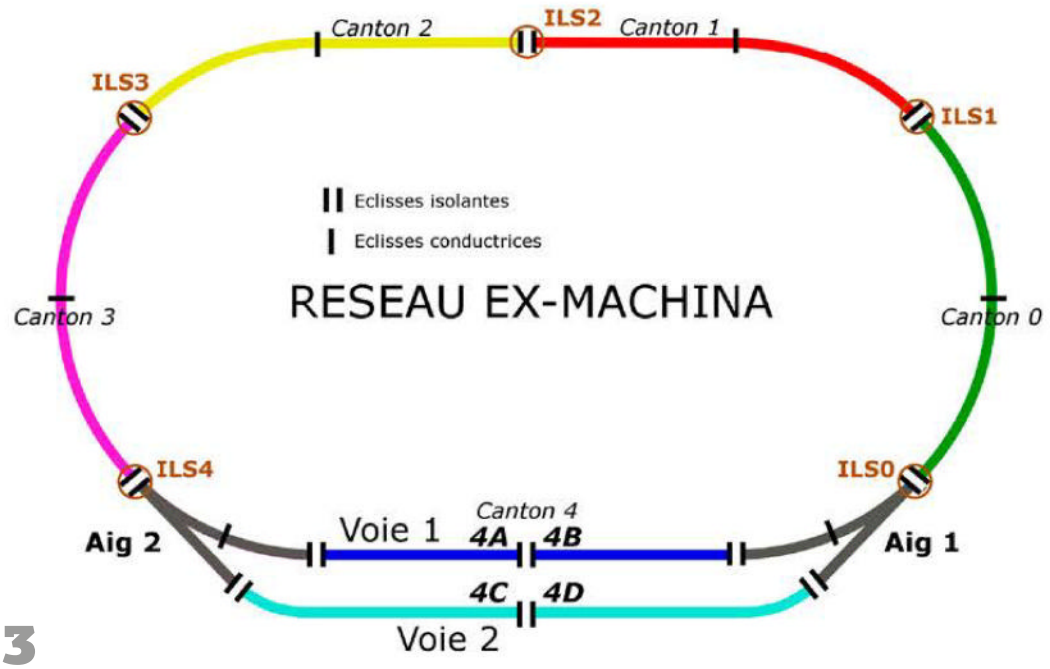
Sur la partie gauche de la **figure 1**, un train (disque orange) se trouve sur le canton 2. Si le train circule dans le sens antihoraire, le canton précédent est le canton 1 sur lequel il faut couper l'alimentation en courant. Mais si le train circule dans le sens horaire, le canton précédent est le canton 3, comme le montre la partie droite de la figure 1. On voit alors que le canton précédent dépend du sens de circulation.

Il n'y a pas que cela. La **figure 2** montre que le canton précédent dépend aussi de la position des aiguilles. La partie gauche de la **figure 2** montre un train (cercle orange) sur le canton 3. Si les aiguilles sont en position déviée, le train qui circule dans le sens horaire passe par la voie 1 et le canton précédent est le canton 4 qu'il faut ne pas alimenter. Mais si les aiguilles sont en position droite, alors le train passe par la voie 2 et le canton précédent est le canton 5, comme le montre la partie droite de la **figure 2**.

Sur notre locodrome, on a simplifié ce problème puisqu'il n'y a qu'un seul canton (le 4) qui est soit la voie 1, soit la voie 2, ceci en fonction de la position des aiguilles. Ainsi, c'est toujours le même numéro de canton qui se trouve entre les cantons 3 et 0. Le fait de connaître la voie active nous permet de savoir sur quels relais agir pour couper l'alimentation du canton 4. Comme on le voit, pour connaître le canton précédent sur lequel il faut couper le courant, on doit considérer l'état exact du réseau et notamment le sens de circulation et la position des aiguilles. Le sens de circulation est donné par la variable sensCircul et la position des aiguilles dépend de la variable voieActive. Quant à la signalisation lumineuse, elle découlera des états d'occupation de tous les cantons comme nous le verrons dans le prochain article.

Repérage de la position des trains : la détection par consommation

Il existe plusieurs techniques pour repérer où sont les trains sur un réseau miniature. Nous allons en voir deux. La première fait appel à des détecteurs d'occupation par consommation de courant, technique suffisamment décrite pour que je ne revienne pas dessus. L'avantage de cette technique est qu'elle délivre un signal pendant toute la durée où le train occupe le canton : on ne risque donc pas de le louper. Évidemment, le signal délivré doit être compatible avec notre carte Arduino Mega, c'est-à-dire être compris entre 0 et 5 V. Le programme qui gère les différents détecteurs est très simple à écrire : si le détecteur réagit, c'est que le canton est occupé et il n'y a aucun calcul à faire. Un autre avantage est qu'on détecte non seulement la locomotive mais le train entier (si les remorques consomment mais ceci est facile à obtenir). L'inconvénient de cette technique est son prix : dans le commerce, on arrive ▸▸



3

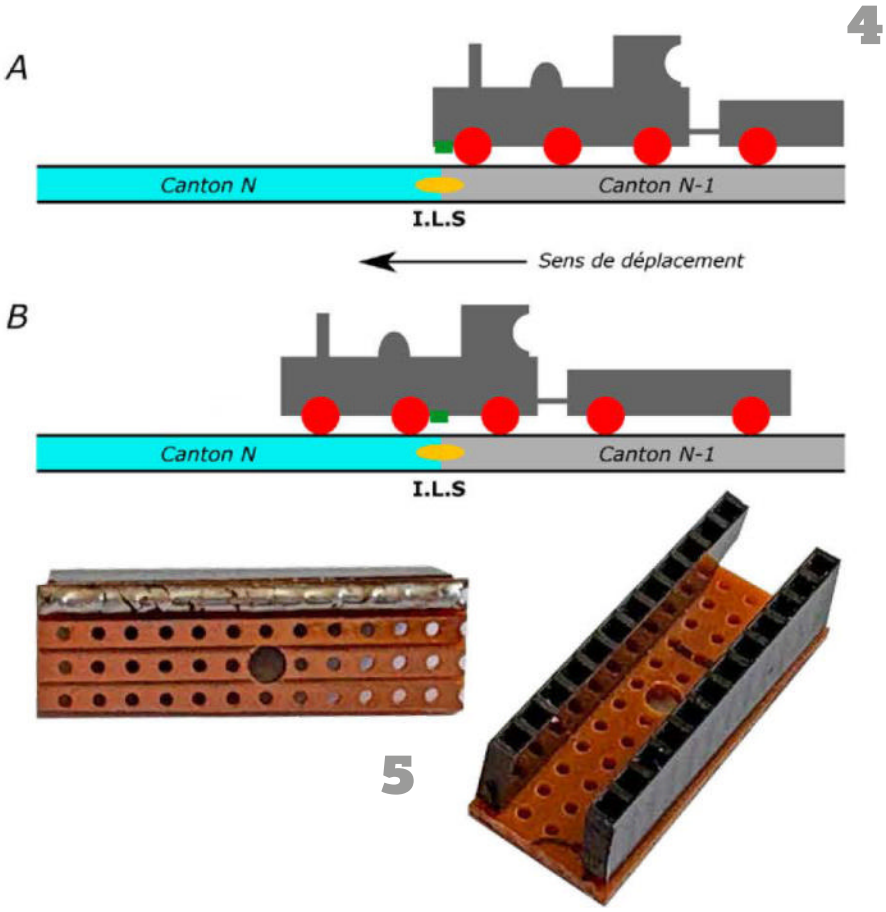
souvent à 10 à 15 euros par détecteur et il nous en faut 8 pour notre locodrome (un par canton ou zone en gare). De plus, ce genre de détecteur est efficace en numérique car la voie est constamment sous tension ; en analogique, le détecteur doit être capable de détecter le train quel que soit le sens de circulation et même si ce dernier est à l'arrêt. Il est alors nécessaire d'injecter sur la voie une tension servant à la détection même au cas où le courant de traction a été coupé. Ceci complique sérieusement les choses pour un réseau analogique.

Repérage de la position des trains : par I.L.S

La deuxième technique fait appel à des I.L.S (interrupteur à lames souples) situés à chaque entrée de canton. Les I.L.S détectent la locomotive qui doit être munie d'un aimant : chaque fois qu'un I.L.S réagit, c'est qu'un train le survole et en conséquence, qu'il entre dans le canton, libérant ainsi le canton précédent. On peut faire mieux en mettant aussi un aimant sur le dernier wagon d'un convoi afin d'être certain que tout le train est bien sorti d'un canton : cette technique qui a été décrite dans la fiche pratique III.88 de LR931 (février 2025), consiste à compter le nombre de déclenchements d'I.L.S en entrée de canton et le comparer avec le nombre de déclenchements en sortie. Elle couvre le cas de la rupture d'attelage et si un wagon reste sur un canton, celui-ci reste occupé jusqu'à ce qu'un opérateur intervienne. C'est cette technique qui a été retenue pour le réseau EX_MACHINA car son prix est bon marché et qu'elle ne demande aucune électronique supplémentaire. Son inconvénient est que le signal délivré est fugace et qu'il faut scruter les capteurs en permanence pour ne pas le louper : ceci exclut l'utilisation de la fonction delay () qui est bloquante, sauf pour de très courtes durées, et il faut bannir toute portion de programme qui met un certain temps à s'exécuter. Dieu merci pour nous, les trains ne se déplacent pas si vite sur le réseau et un programme bien conçu ne devrait pas empêcher la détection des déclenchements d'I.L.S. S'il ne faut pas louper un événement, il ne faut pas non plus en compter plusieurs pour un seul survol ; en effet, un I.L.S est un interrupteur mécanique qui va provoquer des rebonds du signal qui peuvent être vus comme plusieurs déclenchements ; de petites temporisations peuvent résoudre ce problème. Comme pour les poussoirs, les I.L.S sont placés entre la masse (GND) et les entrées A8 à A12 qui sont déclarées en INPUT_PULLUP ; chaque fois qu'un I.L.S est déclenché, l'entrée sur laquelle il est connecté passe à l'état LOW. La **figure 3** montre où il faut placer les I.L.S sur le réseau ; ils sont positionnés à la limite entre cantons, ils sont numérotés à partir de 0 dans le sens antihoraire, et l'I.L.S 1 est à l'entrée du canton 1 si le sens de circulation est antihoraire et à l'entrée du canton 0 s'il est horaire. Il faut cinq I.L.S et les I.L.S 0 et 4 servent aussi à repérer l'entrée en gare.
La **figure 4** montre comment placer l'aimant (représenté en vert) sous la locomotive : celui-ci ne doit pas être trop en avant. En effet, l'I.L.S du canton N se déclenche dès qu'il est survolé par l'aimant, ce qui coupe l'alimentation du canton (N-1). La partie A de la figure montre que si l'aimant est trop à l'avant, les roues qui captent le courant sont encore toutes sur le canton (N-1) quand on coupe le courant et la locomotive va s'arrêter alors qu'elle devrait continuer. La partie B de la figure montre que si l'aimant est placé vers le milieu, les roues qui captent le courant seront déjà sur le canton N quand le canton (N-1) est coupé, ce qui permet à la locomotive de continuer sa course.

Interface GND-5V

Une carte Arduino Mega propose 5 connexions pour la masse (GND) et seulement 3 pour le +5 V. Or il y a déjà 5 I.L.S à relier au GND, sans compter les cartes relais, le tableau de commande et dans l'avenir, la signalisation lumineuse. Avec des signaux à cathodes communes, le décompte donne 4 connexions au 5 V et 18 au GND. Avec des signaux à anodes communes, le décompte donne 12 connexions au 5 V et 10 au GND. Il est donc nécessaire de prévoir une petite interface GND-5V, un peu comme une multiprise sur laquelle on peut se raccorder. Quand on se raccorde à cette interface, il convient de ne pas dépasser la capacité totale de ce que peut délivrer la carte Mega, un peu comme on le ferait avec une multiprise où la puissance totale des appareils raccordés ne doit pas dépasser une certaine valeur. La **figure 5** montre cette interface où il suffit de souder deux connecteurs femelles pour câbles



Dupont sur un morceau de circuit imprimé à bande cuivrée (chaque connecteur sur une même bande). Il est aussi possible de prévoir une alimentation DC 5 V à part pour alimenter la carte 8 relais et les servomoteurs, ce qui permet de garder le 5 V de la carte Arduino pour la signalisation lumineuse et les capteurs (tableau de commande, I.L.S). Dans ce cas, la masse de l'alimentation doit être reliée à la masse de la carte Arduino.

Installation des I.L.S figure 6

Chaque I.L.S est doté de deux fils, un relié au GND de notre interface GND-5V, l'autre à une des entrées A8 à A12. Les pattes des I.L.S sont coudées à 90° et deux trous sont percés au milieu de la voie pour enfiler fil et broche des I.L.S. Les fils sont passés sous le réseau pour ressortir à proximité de l'interface GND-5V. La figure 6 montre un I.L.S collé sur les traverses de la voie.

Programme de gestion des cantons

Chaque canton doit avoir son compteur qui est incrémenté par l'I.L.S d'entrée de canton et décrémenté par l'I.L.S de sortie. La règle est alors simple : un canton est libre si son compteur est égal à 0 et occupé sinon. C'est le sens de circulation qui détermine pour un I.L.S quel canton doit être incrémenté et quel canton doit être décrémenté. De plus, un canton est soit libre, soit occupé : il n'y a que deux états possibles. Son état peut alors être mémorisé dans une variable booléenne qui vaut « true » si le canton est occupé et « false » s'il est libre. Toutes ces règles doivent être expliquées à l'IA sans rien oublier et le mieux est de lui imposer les noms et les types de variables à utiliser :

```
// Definition des 5 ILS
// --------------------
const byte pinILS [] = {A8, A9, A10, A11, A12};

// Compteur des cantons
// --------------------
int compteurCanton [] = {0, 0, 0, 0, 0};

// Les etats d'occupation des 5 cantons
// ------------------------------------
bool occupationCanton [] = {false, false, false, false, false};
```

Une boucle permet de scruter tous les I.L.S avec un indice qui varie de 0 à 4 (5 I.L.S !). Si un I.L.S est déclenché, l'entrée est à l'état LOW. On connaît l'indice de l'I.L.S déclenché, et avec le sens de circulation, on peut déduire quel est le canton concerné pour incrémenter son compteur et quel est le canton précédent pour décrémenter son compteur. Mettre à jour le tableau occupationCanton n'est pas compliqué et couper l'alimentation sur les cantons concernés non plus, mais il faut prendre en compte le sens de circulation.

Pour faire tout cela, je dois reconnaître que l'IA a été bien plus performante que moi. Ma méthode était de considérer chaque canton car il y en a peu. Sa méthode a été de déterminer un algorithme, ce qui est bien plus performant surtout si ce traitement devait être exporté sur un réseau plus complexe. Le plus difficile pour vous sera d'expliquer à l'IA le principe de détection par I.L.S, mais une fois qu'elle aura compris, je suis certain qu'elle vous étonnera ! Compléter votre simulateur Wokwi pour essayer les nouvelles fonctions écrites par l'IA : les I.L.S doivent être remplacés par des poussoirs puisqu'ils n'existent pas comme composants. Le tableau de commande vous permet déjà de sélectionner un train et son sens de circulation : en cliquant sur les poussoirs des I.L.S (dans le bon ordre), vous simulez son avancement et vous pouvez observer comment les cantons sont alimentés. Bien évidemment, l'essai des programmes directement sur le locodrome est fortement recommandé.

Conclusion

Cet article était sans doute le plus difficile de cette nouvelle série. Il faut comprendre les principes de localisation des trains et de traitement des cantons pour communiquer avec l'IA. Celle-ci ne peut répondre qu'à vos demandes ; il est donc important, comme je l'ai déjà dit, de bien prompter avec elle. Ensuite, c'est logiquement elle qui fait le travail. Plus vous essaierez et plus vous serez capable de comprendre ce qu'elle vous fournit, comment elle écrit son code, pourquoi elle choisit de faire ainsi plutôt qu'autrement. C'est pour vous une façon de progresser très vite en programmation d'Arduino. Dans le prochain article, nous traiterons la signalisation lumineuse car pour l'instant, notre B.A.L est un block automatique mais il n'a rien de lumineux ! ■