Nombre: _____ Clase: ___ Fecha: _____

Tarea n.º 1

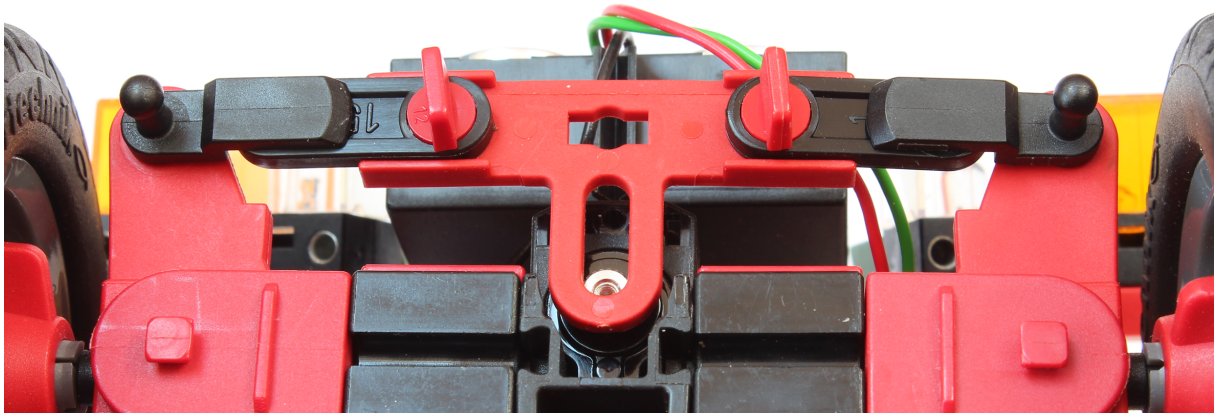
Tacómetro, odómetro y taxímetro

Montar el vehículo y equiparlo con un tacómetro (indicador de velocidad) y un odómetro (cuentakilómetros). Adicionalmente, puedes equiparlo con un taxímetro.

Tarea de construcción

Monta el vehículo como indica el manual de instrucciones. En primer lugar, solo conecta el motor (M1) y el codificador (C1 y la salida de tensión de 9 V) al TXT (véase el esquema de conexiones). En esta tarea, se equipará el vehículo con un tacómetro (indicador de velocidad), un odómetro (cuentakilómetros) y un taxímetro (calculadora de tarifas).

Observa: En principio, aún no deberías colocar la palanca del servomotor; la próxima tarea se ocupa del desplazamiento en línea recta. Es más sencillo colocar la lengüeta agarradera de servo al revés para bloquear la dirección:



Montage Servo-Lasche.jpg

Consejo: Para probar el programa es conveniente, en primer lugar, «levantar sobre tacos» el vehículo, es decir, colocar algo por debajo del vehículo o del parachoques trasero, de modo que las ruedas traseras no tengan contacto con el suelo.

Tareas de programación

1. Visualización de los valores del codificador

En primer lugar, diseña un programa que encienda el motor y, a continuación, muestre continuamente en la pantalla TXT los impulsos del codificador (C1) contados desde el inicio.

2. Odómetro

A partir de los impulsos del codificador, ahora debes calcular la distancia en metros recorrida por el vehículo y mostrarla en la pantalla del TXT (odómetro o cuentakilómetros).

2a. Para ello, primero debes determinar la relación de transmisión del sistema y medir la circunferencia de los neumáticos (véase también la tarea n.º 6 del Robotics TXT 4.0 Base Set). ¿Qué valores obtienes? Determina una fórmula para convertir los impulsos del codificador en la distancia recorrida (en metros).

2b. Diseña un programa de prueba sencillo con el que puedas comprobar la precisión de la medición a lo largo de, por ejemplo, un trayecto de prueba de 2 m. En caso necesario, corrige el factor de conversión.

2c. Adapta tu programa Blockly de la tarea de programación n.º 1 y la salida de la pantalla de modo que se visualice la distancia recorrida.

2d. La entrada del contador puede recibir un valor máximo de 65 535. ¿Cuál es la distancia máxima (en metros) que puede medir su cuentakilómetros?

3. Tacómetro

Además de la distancia recorrida, en la pantalla del TXT ahora también debería visualizarse velocidad actual del vehículo (tacómetro).

3a. Para determinar la velocidad es necesario calcular la distancia recorrida en una unidad de tiempo fija (por ejemplo, en un segundo). Determina una fórmula para calcular la velocidad.

3b. Amplía el programa Blockly desde la salida de programación 2 para calcular la velocidad y visualizar la velocidad actual en la pantalla del TXT además de la distancia recorrida.

Tareas experimentales

1. Tacómetro y odómetro concurrentes

La visualización de la distancia recorrida y la velocidad debe realizarse en un programa concurrente (*Thread*) paralelo al programa de control del vehículo.

Rediseña el programa de modo que la visualización se realice en paralelo al programa principal y se actualice continuamente. El motor debe encenderse en el programa principal.

2. Taxímetro

Para poder utilizar el vehículo autónomo después como taxi, ahora se incorpora un taxímetro. Monta un pulsador en el vehículo y conéctalo a I8.

Al presionar el pulsador por primera vez, el taxímetro debe ponerse en marcha y mostrar la tarifa en la pantalla del TXT. Al presionar el pulsador una segunda vez, el taxímetro debe detenerse y mostrar la tarifa a pagar.

2a. El precio se compone de una tarifa básica y una tarifa que se calcula a partir de la distancia recorrida. Fija una variable para la tarifa básica y la tarifa sujeta al recorrido.

2b. Dibuja un diagrama de transición de estados para el taxímetro.

2c. Amplía el programa Blockly como corresponde. La tarifa básica y la tarifa sujeta al recorrido también deben visualizarse en la pantalla.

Anexos

Tarea n.º 1: Tacómetro, odómetro y taxímetro

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.
- Cinta métrica o metro plegable, tiras de papel (para medir la circunferencia del neumático).

Más información

[1] Andreas Wolf: *Tachometer. Die Geschichte eines unverzichtbaren Instruments*. 04/08/2014.

[2] Alper Aribal (SeoRocket): *Taxameter*. DeWiki.de.

Tarea n.º 1: Tacómetro, odómetro y taxímetro

En esta tarea, en primer lugar se construye un vehículo con tracción trasera y mangueta de dirección (servomotor). Como preparación para el mando autónomo se han programado un tacómetro (con indicación de la velocidad) y un odómetro (indicación de la distancia recorrida). Finalmente, el odómetro puede convertirse en un taxímetro.

Tema

Construcción de vehículos, evaluación del codificador para determinar la velocidad, cálculo de tarifas y procesos concurrentes (hilos).

Objetivos de aprendizaje

- Construcción de vehículos: Mangueta de dirección, accionamiento diferencial
- Conversión de los impulsos del codificador en distancias (relación de transmisión, circunferencia de la rueda), cálculo de tarifas
- Conversión de los impulsos del codificador en velocidades
- Procesos concurrentes (hilos)

Tiempo necesario

El vehículo autónomo se monta como indica el manual de instrucciones. Dependiendo de la experiencia de los alumnos con la fischertechnik son necesarias una o dos clases (45-90 minutos).

Para el desarrollo del programa para resolver la tarea de programación, las alumnas y los alumnos necesitan una o dos clases (45-90 minutos). En caso necesario, debe proporcionarse asistencia para convertir los impulsos del codificador.

Las alumnas y los alumnos más experimentados pueden adicionar un taxímetro a la tarea experimental. La resolución de esta tarea experimental no es necesaria para comprender las demás tareas.

Anexos
Tarea n.º 1: Tacómetro, odómetro y taxímetro

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.
- Cinta métrica o metro plegable, tiras de papel (para medir la circunferencia del neumático).

Más información

[1] Andreas Wolf: *Tachometer. Die Geschichte eines unverzichtbaren Instruments*. 04/08/2014.

[2] Alper Aribal (SeoRocket): *Taxameter*. DeWiki.de.

Nombre: _____ Clase: ___ Fecha: _____

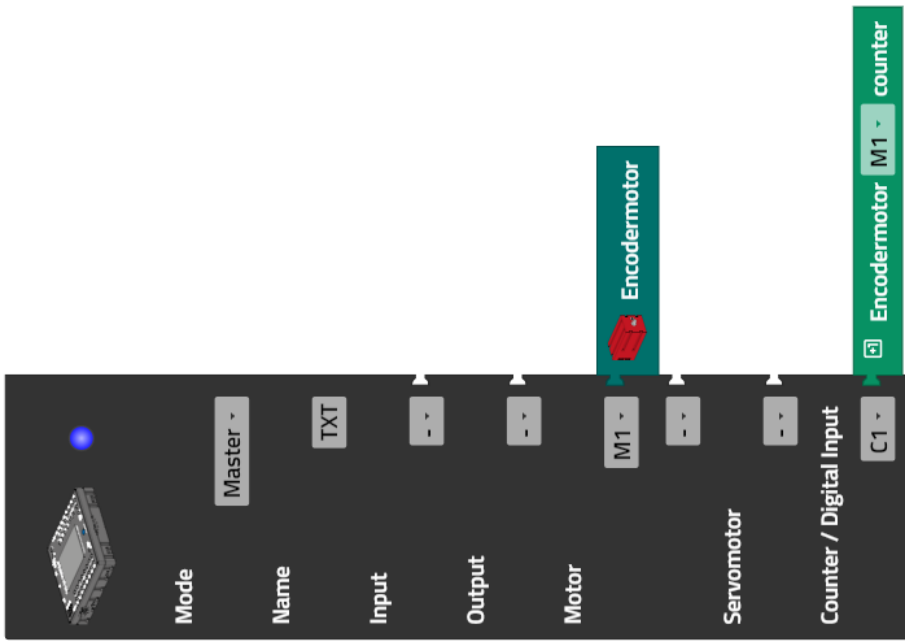
Hoja de soluciones de la tarea temática n.º 1

Tacómetro, odómetro y taxímetro

Con el tacómetro y el hodómetro, la distancia recorrida por impulso debe determinarse primero a partir de la circunferencia del neumático, las señales del codificador y la relación de transmisión del sistema. No es un cálculo difícil, pero es un ejercicio de conversión de unidades. En las siguientes soluciones sugeridas, la información sobre la velocidad se presenta en m/h; sin embargo, también se puede convertir a km/h.

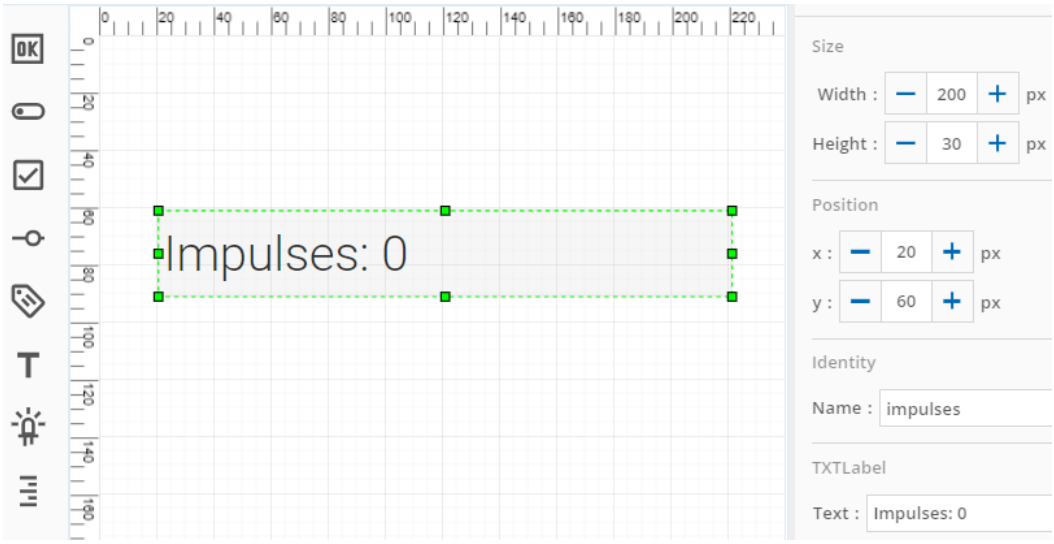
Tareas de programación

Configuración de los actuadores (en esta tarea solo se necesita el motor del codificador):

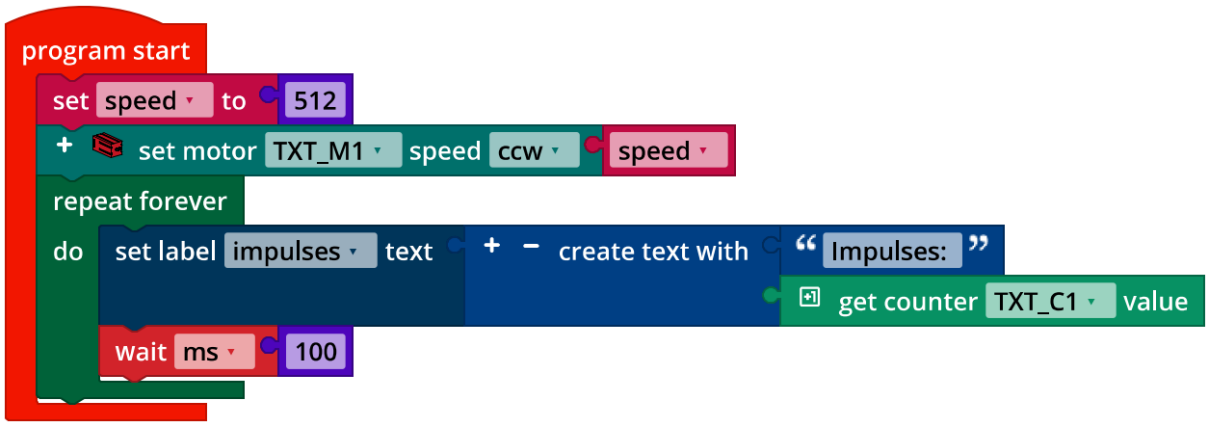


1. Visualización de los valores del codificador

Configuración de pantalla:



Programa (ejemplo) para la visualización de los valores del contador del codificador:



Display_Counter_Encoder.ft

La pausa en el bucle sirve para una indicación controlada y continua en la pantalla. Si se omite, el controlador intenta compensar el retraso a través de la visualización en la pantalla.

2. Odómetro

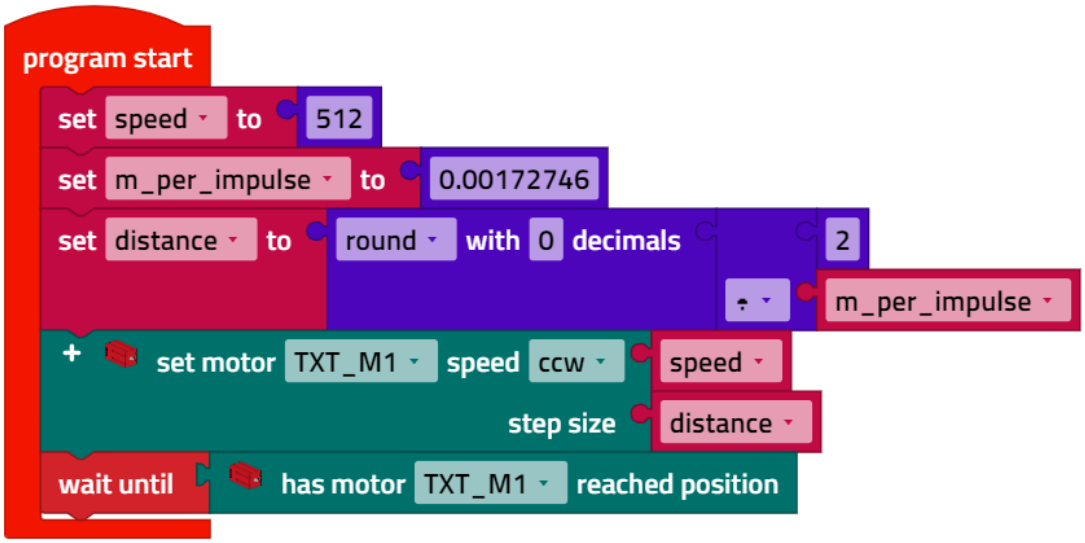
El codificador magnético del motor emite 63,9 impulsos por cada vuelta del eje de salida. La transmisión de la jaula diferencial genera una relación de reducción de 14:26 = 7:13.

2a. La circunferencia del neumático es de aproximadamente 20,5 cm (véase la tarea n.º 6 del Robotics TXT 4.0 Base Set; métodos de medición recomendados: colocar el borde del papel alrededor de los neumáticos, marcar la circunferencia y después medir la longitud del borde del papel hasta la marca).

Por lo tanto, a partir de los impulsos i contados a lo largo de un recorrido, puede medirse la distancia d (en m) según la siguiente fórmula:

$$d = i \cdot \frac{1}{63{,}9} \cdot \frac{7}{13} \cdot 20{,}5 \cdot \frac{1}{100}\ \text{m} \approx i \cdot\ 0{,}00172746\ \text{m}$$

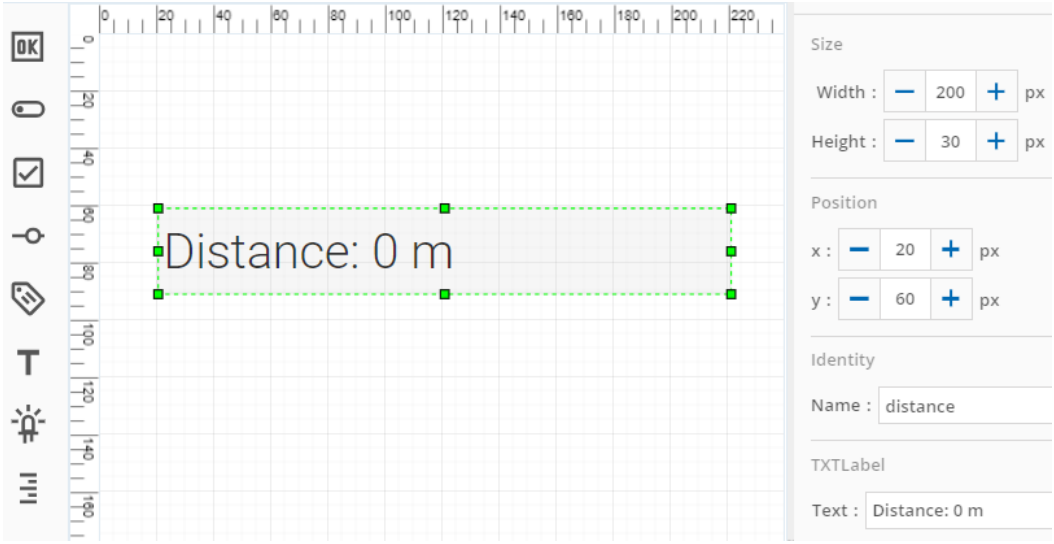
2b. Programa de prueba sencillo para comprobar la medición:



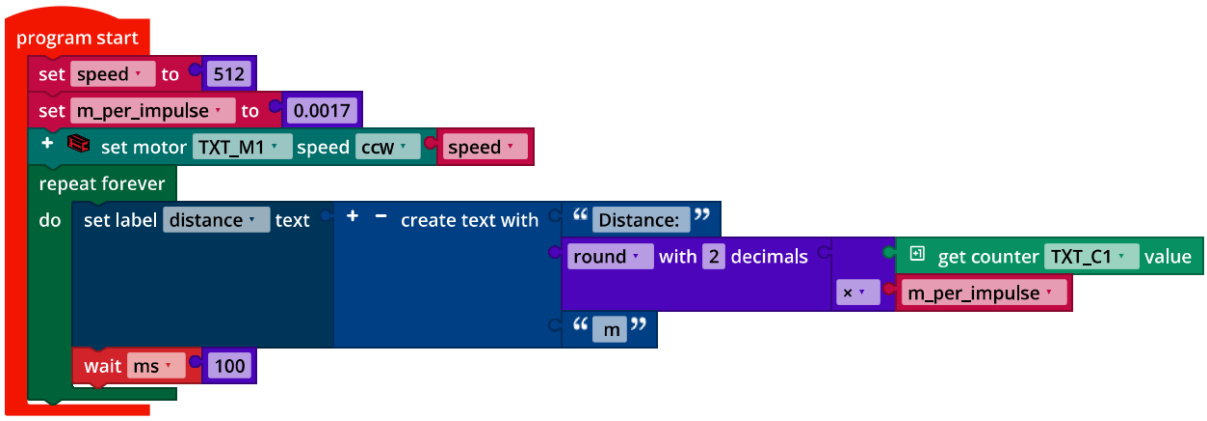
Distance_Test.ft

En las pruebas, el vehículo se detiene un poco antes de alcanzar la marca de 2 m. Por lo tanto, el factor de conversión se corrigió a 0,0017.

2c. Configuración de pantalla:



Programa (ejemplo) del odómetro:



Odometer.ft

2d. La distancia máxima que puede medirse $d_{máx.}$ es de

$$d_{máx.} = 65.535 \cdot 0{,}0017\ \text{m} \approx 111{,}41\ \text{m}$$

3. Tacómetro

Un intervalo de tiempo fijo para determinar la velocidad es, por ejemplo, 1 s: lo suficientemente largo como para obtener una medición precisa y lo suficientemente corto como para reaccionar rápidamente a los cambios de velocidad.

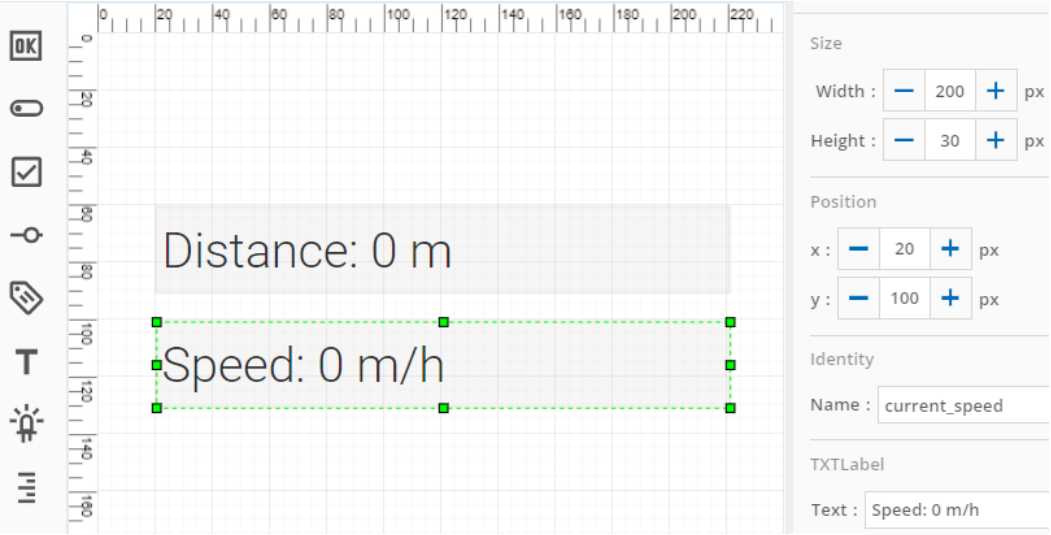
3a. Así es posible medir la velocidad v a partir de los impulsos contados en un segundo i de la siguiente manera:

$$v = i \cdot \frac{1}{63{,}9} \cdot \frac{7}{13} \cdot 20{,}5 \cdot 36 \ {}^{\mathrm{m}}/_{\mathrm{h}} \approx i \cdot 6{,}2189 \ {}^{\mathrm{m}}/_{\mathrm{h}}$$

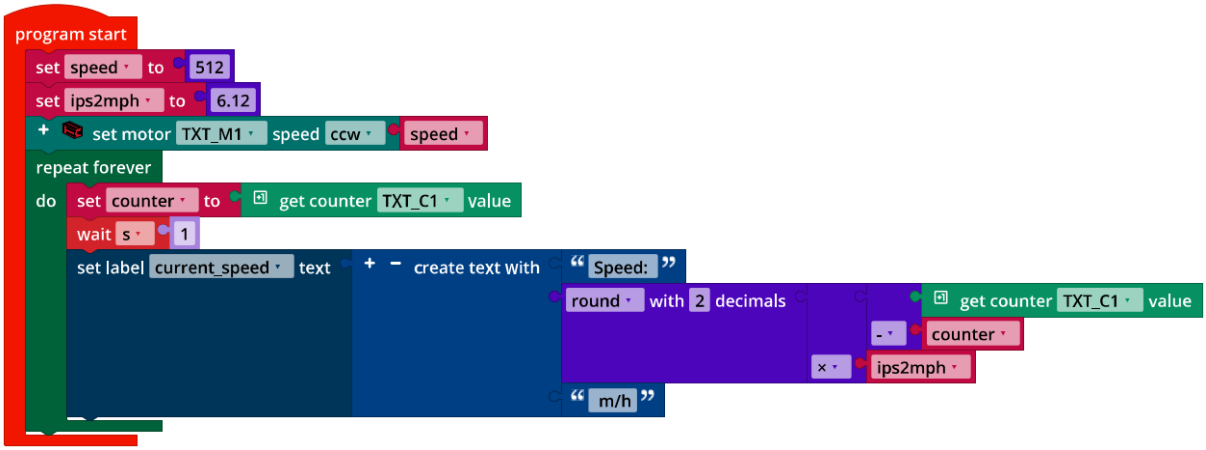
Si consideramos el valor corregido de la tarea de programación n.º 2, resulta:

$$v = i \cdot 0{,}0017 \cdot 3600 \ {}^{\mathrm{m}}/_{\mathrm{h}} \approx i \cdot 6{,}12 \ {}^{\mathrm{m}}/_{\mathrm{h}}$$

3b. Configuración de pantalla:

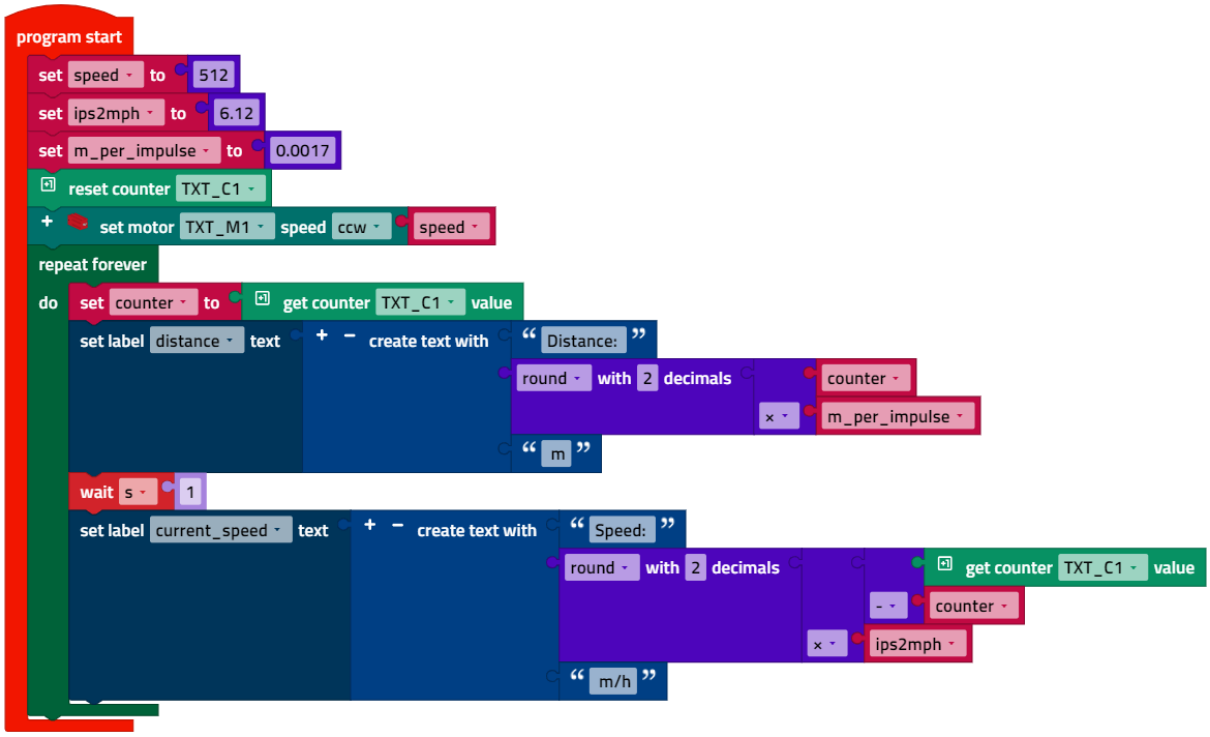


Programa (ejemplo) del tacómetro:



Tachometer.ft

Programa (ejemplo) de odómetro y tacómetro:

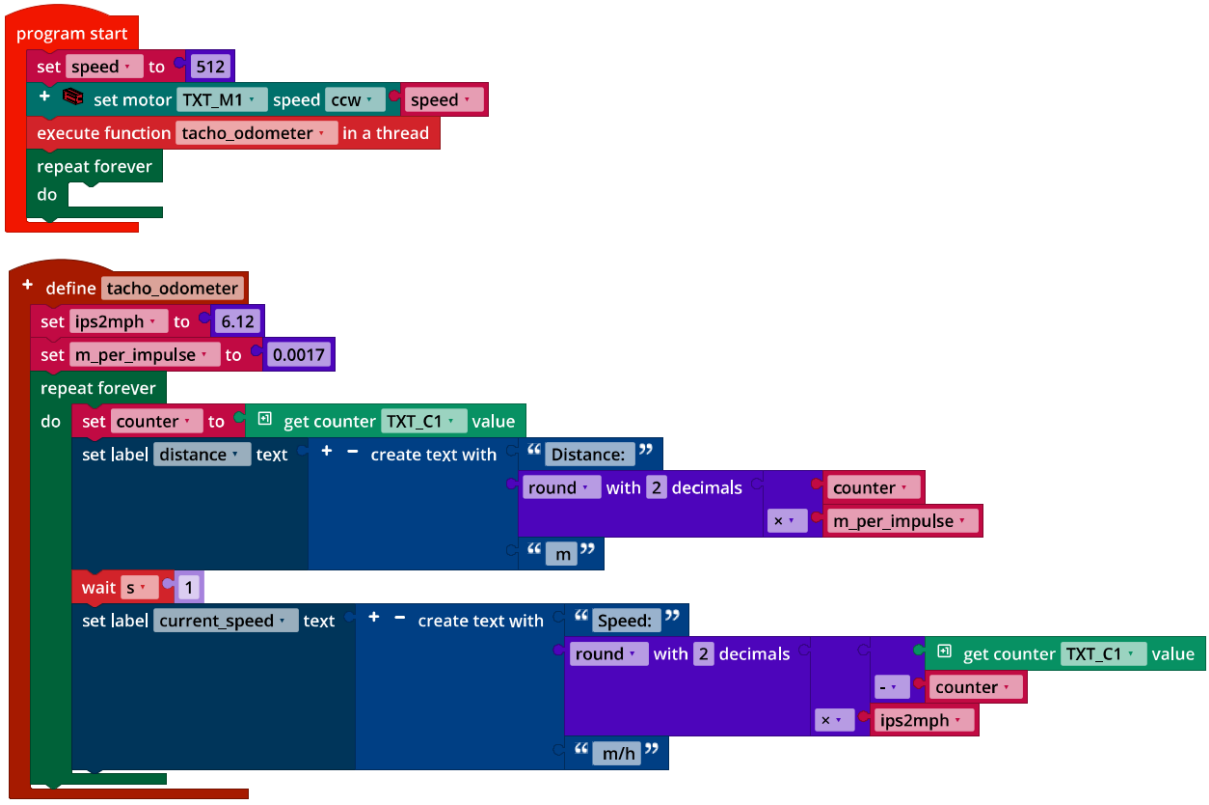


Tachometer_and_Odometer.ft

Tareas experimentales

1. Tacómetro y odómetro concurrentes

Programa (ejemplo):

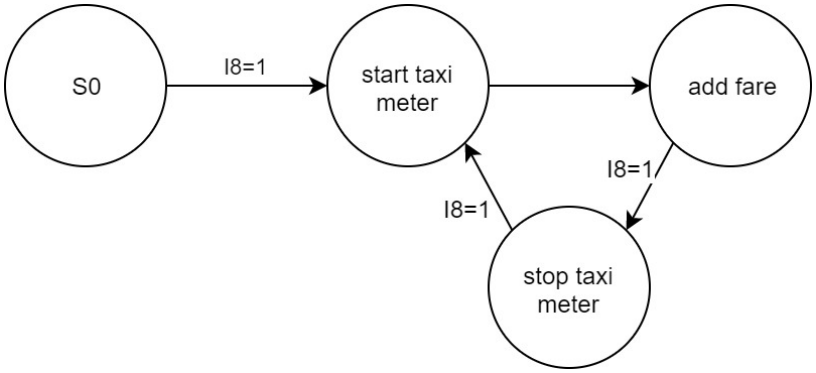


Tachometer_and_Odometer_Thread.ft

2. Taxímetro

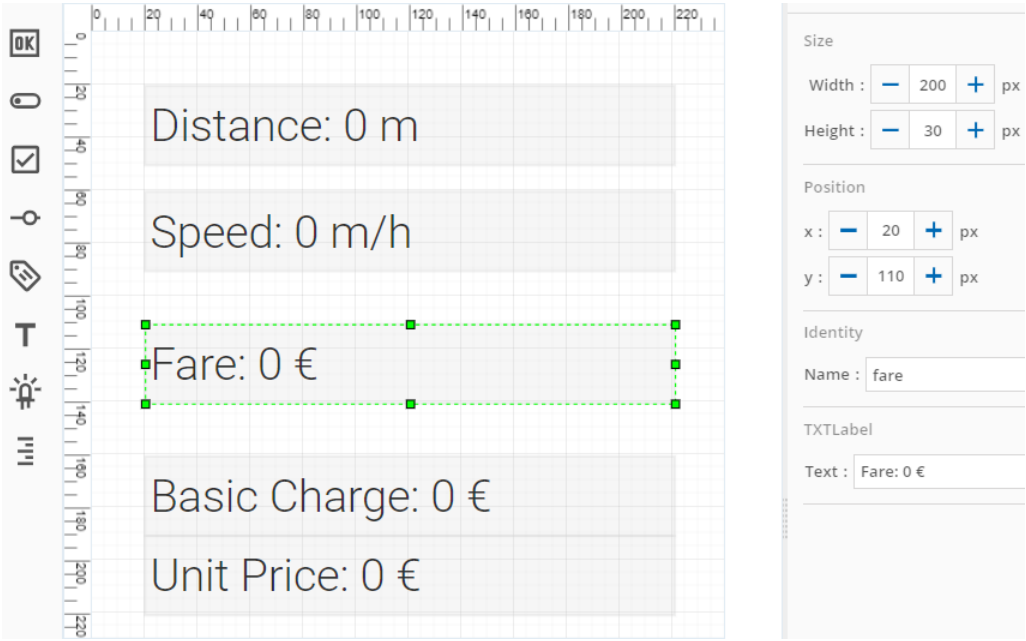
2a. Dado que la relación entre el tamaño del modelo de vehículo y el de un vehículo real es de aproximadamente 1:10, un taxímetro es apropiado para indicar un precio para distancias de 100 m, por ejemplo, 2 euros/100 m y 3,50 euros de tarifa básica.

2b. Diagrama de transición de estados:



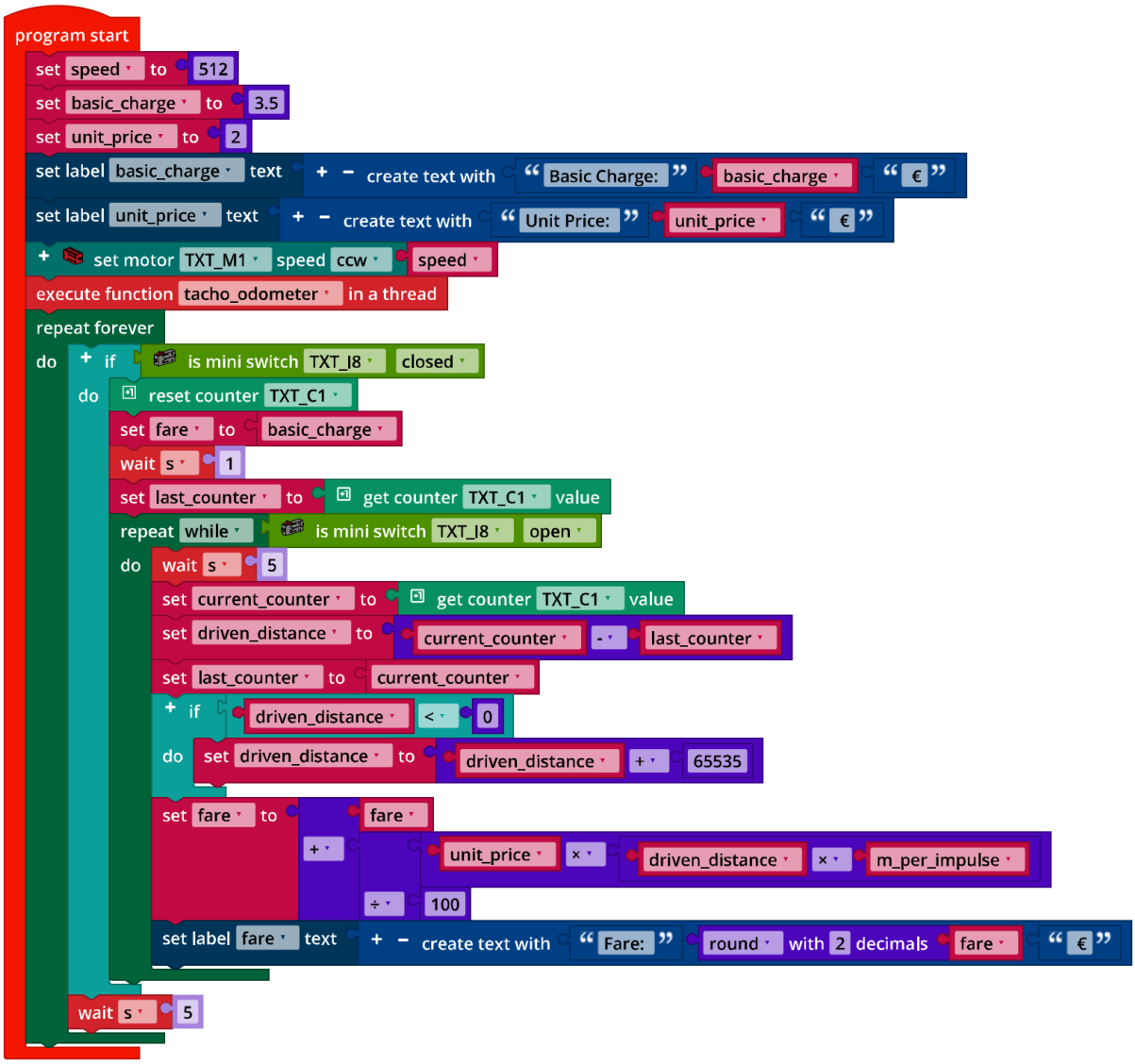
State-Transition_Diagram_Taxi_Meter.drawio

2c. Configuración de pantalla:



Los impulsos del codificador solo se cuentan hasta 65 535. Por lo tanto, la tarifa debe incrementarse regularmente, por ejemplo, una vez cada cinco segundos.

Extracto del programa (ejemplo) del taxímetro:



Taxi_Meter.ft

El taxímetro también puede desplazarse a un proceso concurrente (hilo). En tal caso, al igual que el tacómetro y el odómetro, simplemente «avanza» mientras la navegación del vehículo tiene lugar en el programa principal.

Anexos

Tarea n.º 1: Tacómetro, odómetro y taxímetro

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.
- Cinta métrica o metro plegable, tiras de papel (para medir la circunferencia del neumático).

Más información

[1] Andreas Wolf: *Tachometer. Die Geschichte eines unverzichtbaren Instruments*. 04/08/2014.

[2] Alper Aribal (SeoRocket): *Taxameter*. DeWiki.de.

[3] Editor de diagramas en línea para crear diagramas de transición de estados (formato drawio): https://www.diagrammeditor.de/

Nombre: _____ Clase: ___ Fecha: _____

Tarea n.º 2

Asistente de frenado, control de velocidad constante y asistente de mantenimiento de carril

En esta tarea se desarrollan dos importantes sistemas de asistencia al conductor: frenar a tiempo ante un obstáculo y mantener el carril y la velocidad de forma autónoma. Para garantizar que el vehículo no se aleje de forma incontrolada durante las pruebas, en primer lugar se equipará con una «parada de emergencia».

Tarea de construcción

Para esta tarea necesitamos el servomotor, el sensor ultrasónico y la cámara USB. Conecta el sensor ultrasónico a I1, el servomotor a S1 y la cámara USB a USB1 (véase el esquema de conexiones). Respeta la polaridad correcta de la clavija de enchufe del servomotor: el cable marrón a la izquierda, el naranja a la derecha.

Importante: Cuando se inicia el TXT, el servomotor se coloca automáticamente en «posición línea recta». Coloca la palanca del servomotor de modo que en esta posición las dos ruedas delanteras estén alineadas con suficiente precisión hacia el frente.

Atención: Si mueves el servomotor con la mano cuando el TXT está encendido, puedes dañarlo. Por otro lado, el servomotor no debe controlar una posición que se encuentre más allá del tope de la palanca – por lo tanto, para los comandos del servomotor solo deben utilizarse posiciones entre, aproximadamente, 100 y 400. El rango de dirección de nuestro vehículo es todavía algo menor; se sitúa en valores del servomotor entre, aproximadamente, 130 y 370.

Con la prueba de interfaz puedes medir fácilmente el rango del ángulo de dirección en el que las ruedas delanteras continúan girando libremente.

Ahora equiparemos el coche con un asistente de frenado, un control de velocidad constante y un asistente de mantenimiento de carril.

Tareas de programación

1. Parada de emergencia

Comenzaremos con un mecanismo de «parada de emergencia» antes de hacer que el vehículo conduzca de forma autónoma: El vehículo debe detenerse inmediatamente al oír un aplauso fuerte.

Programa la «parada de emergencia» como un proceso concurrente (hilo) que se controle a través del micrófono de la cámara. A través de la experimentación, puedes determinar un umbral de volumen apropiado en el que el motor debe detenerse, mostrando primero en la consola el valor de volumen determinado por el micrófono.

Observa: El propio motor produce un ruido relativamente fuerte.

2. Desplazamiento en línea recta

Notarás que el vehículo no se desplaza exactamente en línea recta. Antes de alinear automáticamente la dirección con el carril en las tareas experimentales, primero deberías intentar ajustar el servomotor para lograr un desplazamiento en línea recta bastante preciso.

Puedes obtener una primera aproximación con la prueba de interfaz. Sin embargo, es necesario probar el resultado a través de la experimentación: Diseña un programa de prueba Blockly y corrige el valor del ajuste inicial del servomotor hasta que el vehículo mantenga la dirección con la mayor precisión posible en un trayecto de prueba de 2 m.

En los siguientes programas, determina el ajuste correcto del servomotor en una variable y alinea el servomotor con este valor al inicio del programa.

3. Asistente de frenado

El sistema de asistencia de frenado ahora debe garantizar que el vehículo de conducción autónoma no choque contra un obstáculo inmóvil (o en movimiento) si el conductor no está atento o reacciona demasiado tarde. Para ello, es necesario frenar con el tiempo suficiente, de modo que el vehículo no se acerque más de 10 cm al obstáculo.

Puedes medir la distancia a un obstáculo (o a un vehículo que circula por delante) con el sensor ultrasónico que ya conoces del Robotics TXT 4.0 Base Set.

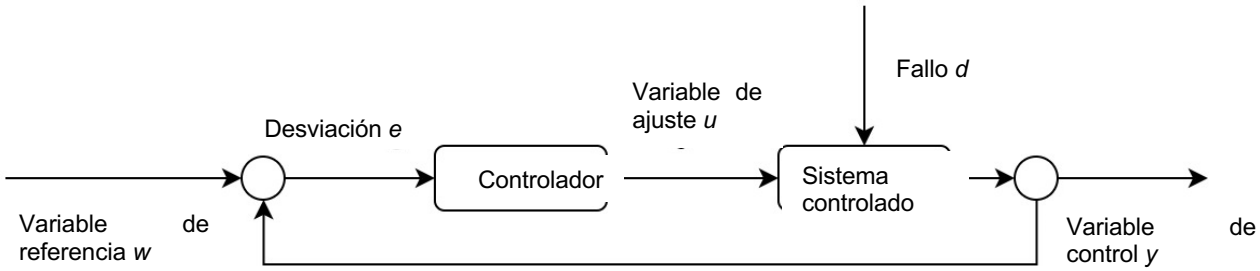
Diseña un programa Blockly adecuado y pruébalo con diferentes obstáculos. Utiliza el hilo de «parada de emergencia» de la tarea de programación n.º 1 para poder detener el vehículo con un aplauso si fuera necesario.

Observa: Tu vehículo también recorre un trayecto durante la medición de la distancia.

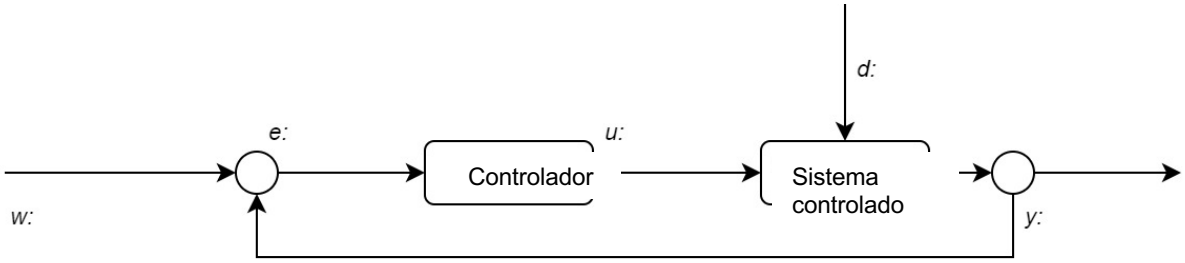
4. Control de velocidad constante

Un sistema de asistencia al conductor incluido actualmente en la mayoría de los coches de tamaño medio es el *control de velocidad constante*: un regulador de velocidad que se ocupa de que el vehículo mantenga una velocidad predeterminada [1].

Para ello, observa primero la siguiente representación de un bucle de control. Ya la conoces de la tarea n.º 8 del Robotics TXT 4.0 Base Set.



4a. ¿Qué valores del control de velocidad constante corresponden a los parámetros *w*, *e*, *u*, *d* e *y*? Etiqueta el siguiente bucle de control como corresponde:



4b. Programa un control de velocidad constante en Blockly. Como variable de referencia debe determinarse la velocidad deseada en m/h.

Consejo: A través del comando «Map» puedes convertir de manera elegante la variable de referencia en la tensión del motor requerida si conoces la velocidad del vehículo en m/h.

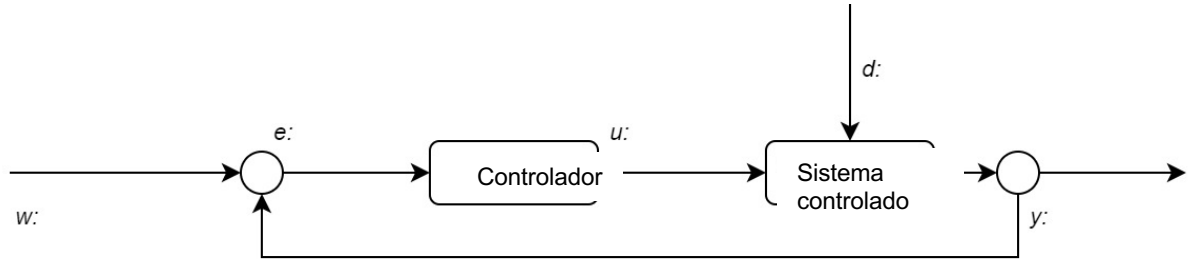
Observa: Después de una «parada de emergencia» o de una parada ante un obstáculo, el control de velocidad constante no debe volver a aumentar la velocidad. Adapta estos dos hilos en tu programa como corresponde.

Tarea experimental

1. Asistente de mantenimiento de carril con controlador P

Ahora el vehículo debe aprender a seguir un carril de forma autónoma. Para ello, debe orientarse con ayuda de la cámara. hacia el límite derecho del carril. También debes implementar este sistema de asistencia al conductor con un controlador proporcional (controlador P).

1a. En primer lugar, etiqueta el siguiente bucle de control:



1b. Ahora añade a tu programa principal un controlador proporcional que corrija la dirección del servomotor en función de la desviación de la pista.

Consejo: Configura la detección de líneas de la cámara de modo que arroje la posición «0» cuando el vehículo se encuentre a una distancia de 2 cm paralelo al límite derecho del carril.

Consejo: Limita la velocidad de tu vehículo a 350 y prueba el controlador P, en primer lugar, con un valor muy pequeño ($k_p = 0{,}1$). Aumente el factor de proporcionalidad hasta que el controlador genere «oscilaciones de amplitud creciente» y después seleccione el valor en el que el regulador se «estabilice» más rápidamente.

Para probar tu controlador, utiliza el tramo de carretera recta de la hoja adjunta.

1c. Añade a tu programa una salida de texto que, después de cada cambio de la posición del límite del carril detectado, proporcione en la consola

- el tiempo (en ms) que ha transcurrido desde el inicio del programa y
- el valor de la posición actual

separados por un espacio.

Después de cada desplazamiento de prueba, copia la información de salida de la consola con un valor diferente de k_p en una hoja de cálculo y representa los valores en un gráfico (x: tiempo, y posición). Selecciona el factor de proporcionalidad k_p, en el que la curva se «estabiliza» más rápidamente.

2. Asistente de mantenimiento de carril con controlador PD

Al igual que con el buggy de la tarea n.º 8 del Robotics TXT 4.0 Base Set, puedes atenuar más el sobreimpulso añadiendo un elemento «D» (componente diferencial) al controlador, que al corregir la dirección considera la magnitud del cambio en la desviación del límite del carril detectado desde la posición «0».

2a. Amplía el controlador P de la tarea experimental n.º 1 a un controlador PD.

2b. Realiza desplazamientos de prueba utilizando valores diferentes para el factor diferencial k_d y representa los datos en un programa de hojas de cálculo.

Consejo: Comienza las pruebas con el factor diferencial $k_d = 0{,}01$ y aumenta progresivamente el valor en 0,005 hasta que el sobreimpulso del controlador P se encuentre bien atenuado.

2c. Si colocas varios tramos de carretera rectos y curvos uno al lado del otro, puedes probar el asistente de mantenimiento de carril en un recorrido desafiante.

Anexos

Tarea n.º 2: Asistente de frenado, control de velocidad constante y asistente de mantenimiento de carril

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.
- Carril con marcas en una hoja adjunta (o el archivo del carril impreso).
- Obstáculo (por ejemplo, un libro o una caja de cartón)

Más información

[1] Jim Meininghaus: *Die Geschichte des Tempomaten. Wie ein Blinder das Autofahren veränderte*. 03/03/2014, motor-talk.de.

[2] Thomas Paulsen: *Autonomes Fahren: Die 5 Stufen zum selbstfahrenden Auto*. 07/11/2018, adac.de.

[3] Editor de diagramas en línea para crear diagramas de transición de estados (formato drawio): https://www.diagrammeditor.de/

Tarea n.º 2: Asistente de frenado, control de velocidad constante y asistente de mantenimiento de carril

El vehículo se equipa con las primeras capacidades autónomas (sistemas de asistencia al conductor). De este modo, debe iniciar la frenada al encontrarse con un obstáculo (asistente de frenado), mantener una velocidad especificada (control de velocidad constante) y seguir un carril de forma autónoma (asistente de mantenimiento de carril).

Tema

Sistemas de asistencia al conductor (asistente de frenado, asistente de mantenimiento de carril y control de velocidad constante), sistema de sensores (evaluación de impulsos del codificador, valores ultrasónicos e imágenes de cámaras para controlar la velocidad, detección de obstáculos y carriles) y controladores (controladores P y PD).

Objetivos de aprendizaje

- Medición de distancia mediante sensor de ultrasonidos
- Control proporcional analógico con medición de la velocidad (control de velocidad constante) y utilizando una cámara con análisis de imágenes (asistente de mantenimiento de carril)
- Controlador PD (asistente de mantenimiento de carril)

Tiempo necesario

El servomotor y la cámara USB también se conectan al vehículo autónomo.

La tarea se centra en el «detector de obstáculos» (medición de distancia por ultrasonido) y el «seguidor de líneas» (controlador proporcional) de las tareas del Robotics TXT 4.0 Base Set. Para el desarrollo del programa para resolver la tarea de programación, las alumnas y los alumnos necesitan 180-270 minutos (entre cuatro y seis clases), dependiendo de la edad y los conocimientos previos. El principio de funcionamiento de un regulador proporcional debería haberse tratado anteriormente en clase.

Las tareas experimentales (asistente de mantenimiento de carril con controladores P y PD) pueden destinarse a las alumnas y los alumnos especialmente capacitados o mayores. La resolución requiere de aprox. 90-180 minutos (entre dos y cuatro clases).

Anexos
Tarea n.º 2: Asistente de frenado, control de velocidad constante y asistente de mantenimiento de carril

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.
- Carril con marcas en una hoja adjunta (o el archivo del carril impreso).
- Obstáculo (por ejemplo, un libro o una caja de cartón)

Más información

[1] Jim Meininghaus: *Die Geschichte des Tempomaten. Wie ein Blinder das Autofahren veränderte*. 03/03/2014, motor-talk.de.

[2] Thomas Paulsen: *Autonomes Fahren: Die 5 Stufen zum selbstfahrenden Auto*. 07/11/2018, adac.de.

[3] Editor de diagramas en línea para crear diagramas de transición de estados (formato drawio): https://www.diagrammeditor.de/

Nombre: _____ Clase: ___ Fecha: _____

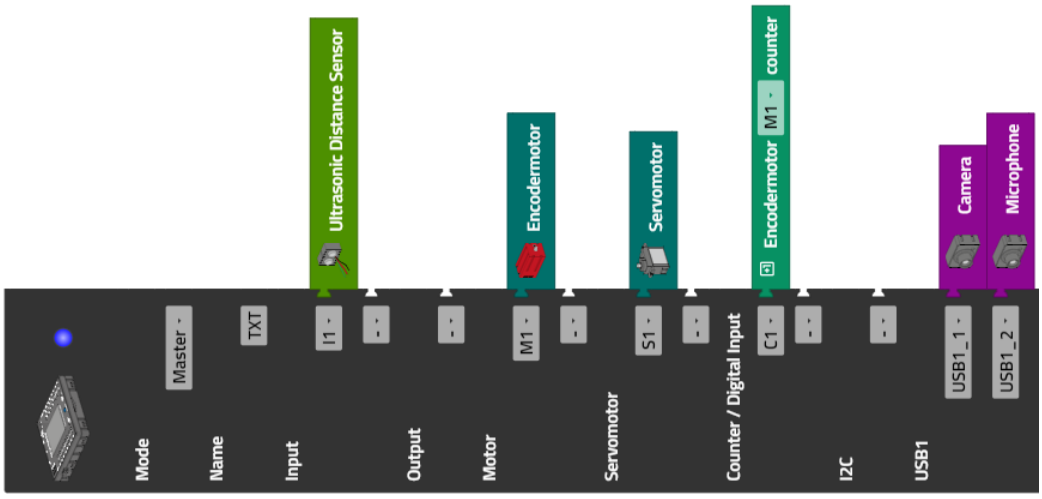
Hoja de soluciones de la tarea temática n.º 2

Asistente de frenado, control de velocidad constante y asistente de mantenimiento de carril

Al programar, las alumnas y los alumnos deberían hacer un amplio uso de la información de salida de las variables de la consola y realizar pruebas, en primer lugar, con un vehículo «elevado sobre tacos» (o sin las ruedas traseras montadas). Las pruebas «en vivo» pueden realizarse utilizando tramos de carriles incorporados en un recorrido. Los tramos de carriles adicionales para imprimir están disponibles en formato PDF para descargar.

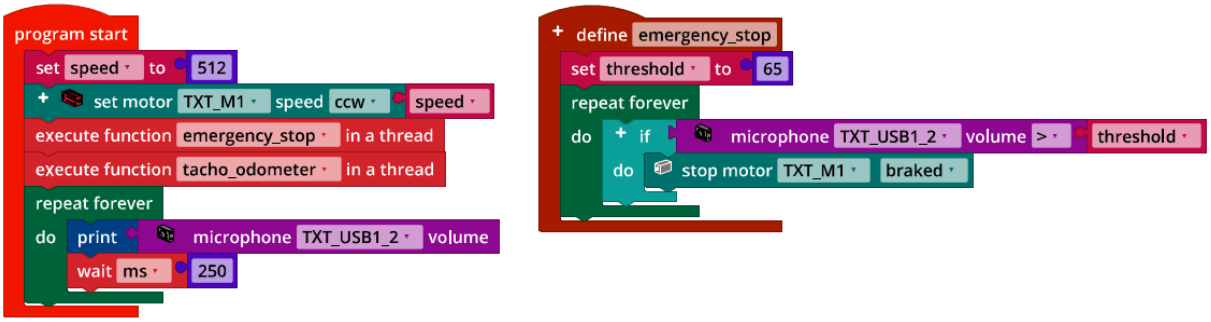
Tareas de programación

Configuración de sensores y actuadores:



1. Parada de emergencia

Extracto del programa (ejemplo): Parada de emergencia del vehículo en caso de que se produzca un ruido fuerte (por ejemplo, un aplauso); visualización del volumen en la consola:

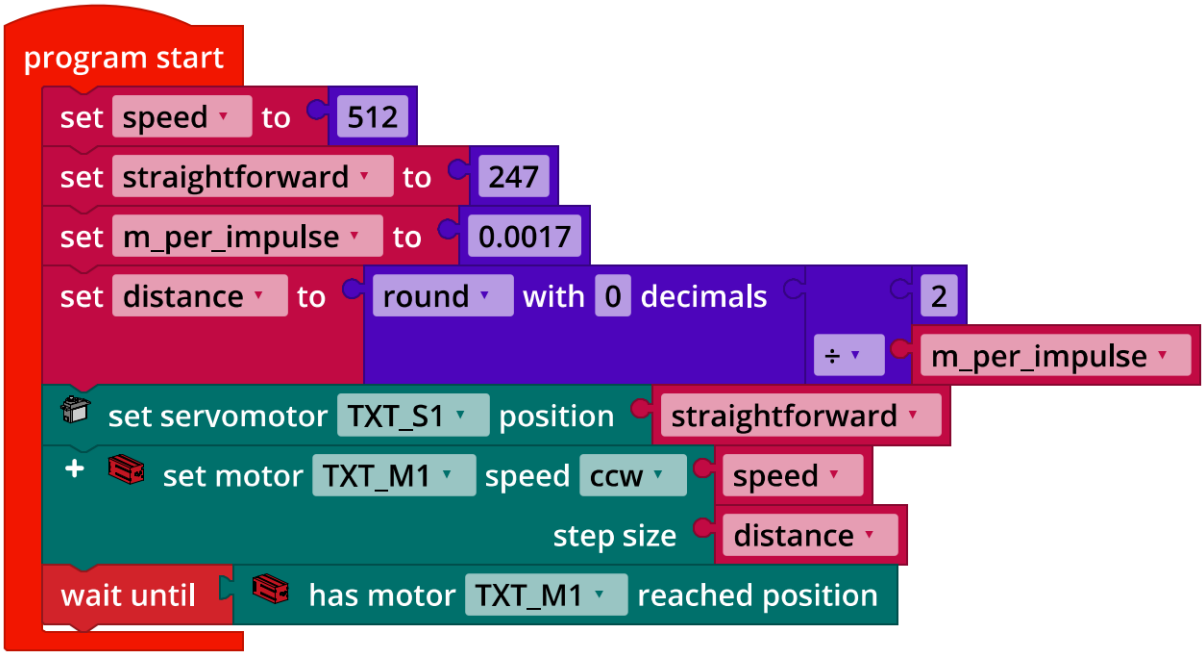


Emergency_Stop.ft

2. Desplazamiento en línea recta

El ajuste del servomotor para el desplazamiento en línea recta varía de un modelo a otro. Puede ser necesario comprobar oportunamente el valor y, eventualmente, reajustarlo. En el siguiente ejemplo de programa para la prueba (experimental) de desplazamiento en línea recta, se determina en la variable «straightforward».

Programa (ejemplo):



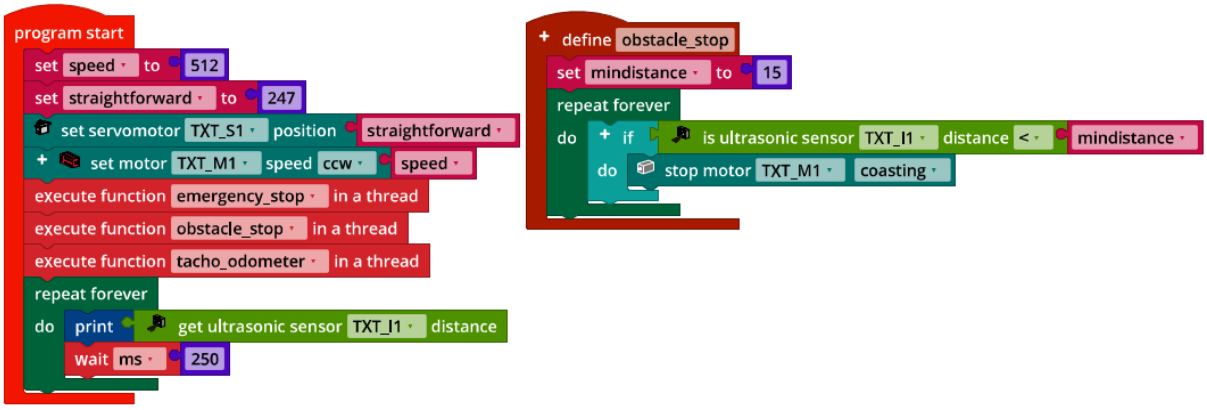
Servo_Calibration.ft

3. Asistencia de frenado

Para que el vehículo frene con el tiempo suficiente, el valor límite de la distancia a la que debe iniciarse el frenado debe ser superior a 10 cm (tiempo de reacción de la medición ultrasónica, distancia de frenado). La distancia de frenado puede simularse deteniendo el motor en modo «coasting».

Para determinar el valor límite de distancia adecuado para iniciar el frenado, es necesario dejar que el vehículo se dirija a máxima velocidad hacia un obstáculo inmóvil y se detenga. El valor límite se ajusta hasta que el vehículo se detenga a 10 cm del obstáculo.

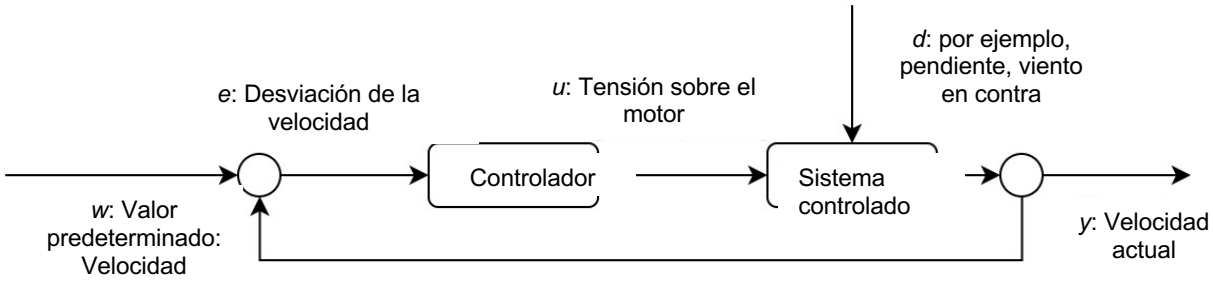
Extracto del programa (ejemplo) del sistema de asistencia de frenado:



Brake_Assist.ft

4. Control de velocidad constante

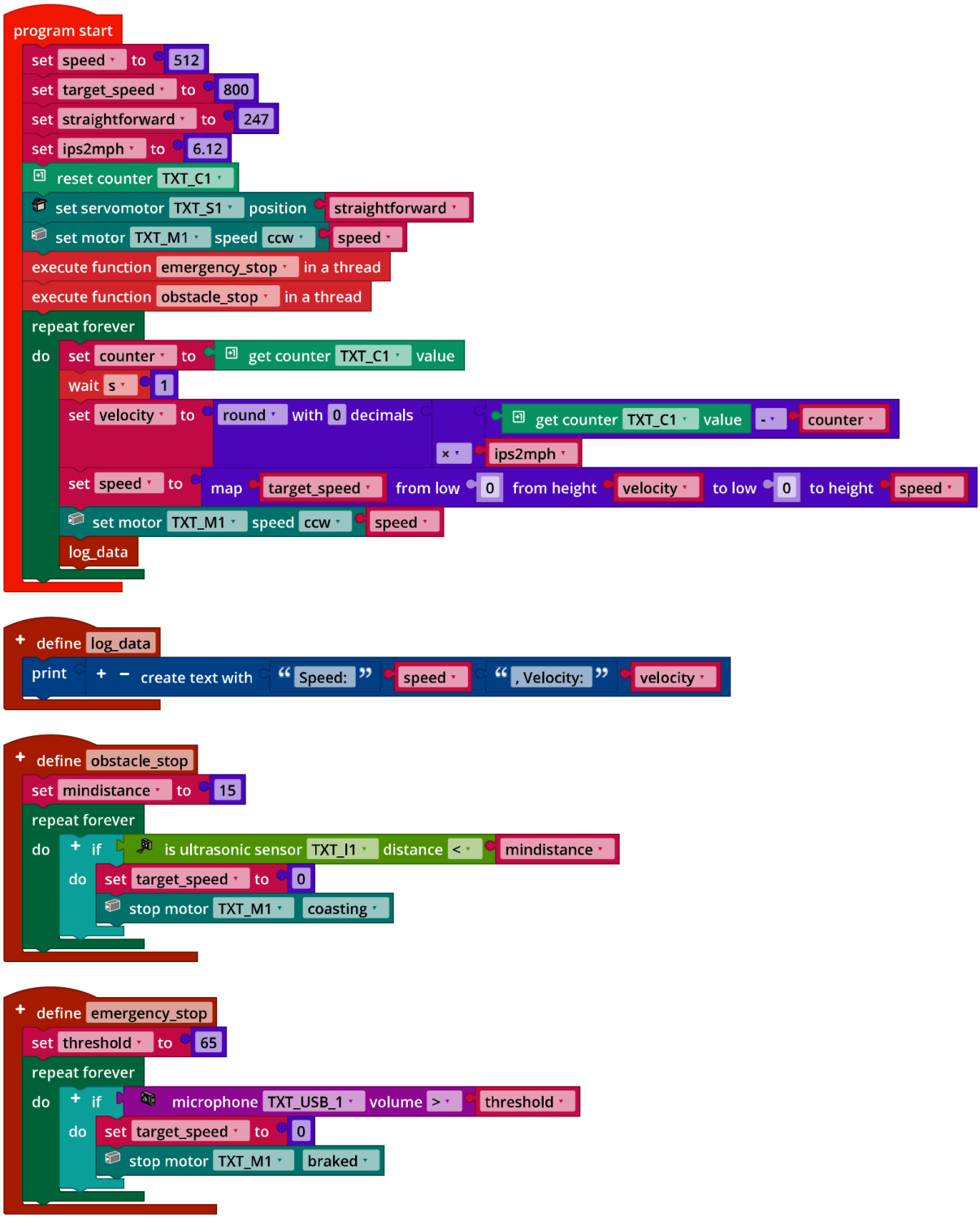
4a. Bucle de control del control de velocidad constante:



Regelkreis_Tempomat.drawio

El control de velocidad constante mantiene la velocidad incluso en una pendiente o ante otras resistencias. Esto puede demostrarse claramente frenando los neumáticos con la mano cuando el vehículo está elevado sobre tacos.

4b. Extracto del programa (ejemplo) del control de velocidad constante:

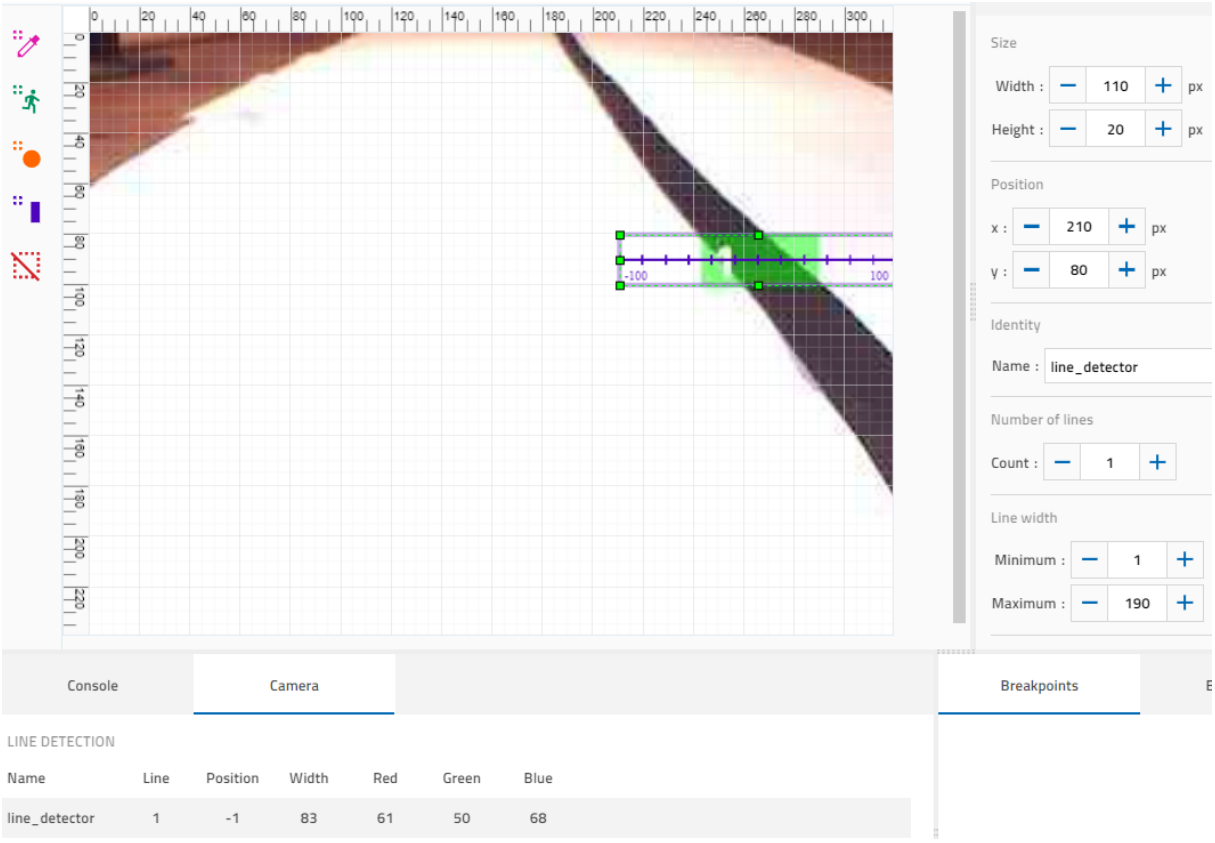


Speed_Control.ft

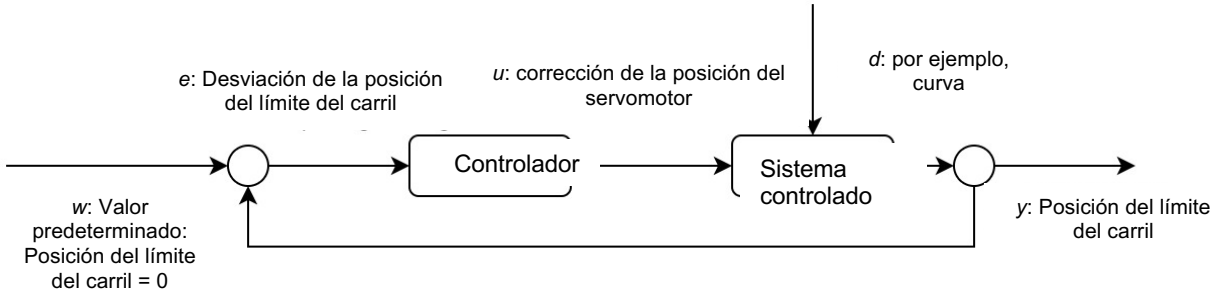
Tareas experimentales

1. Asistente de mantenimiento de carril con controlador P

Configuración de la cámara (detección de líneas):

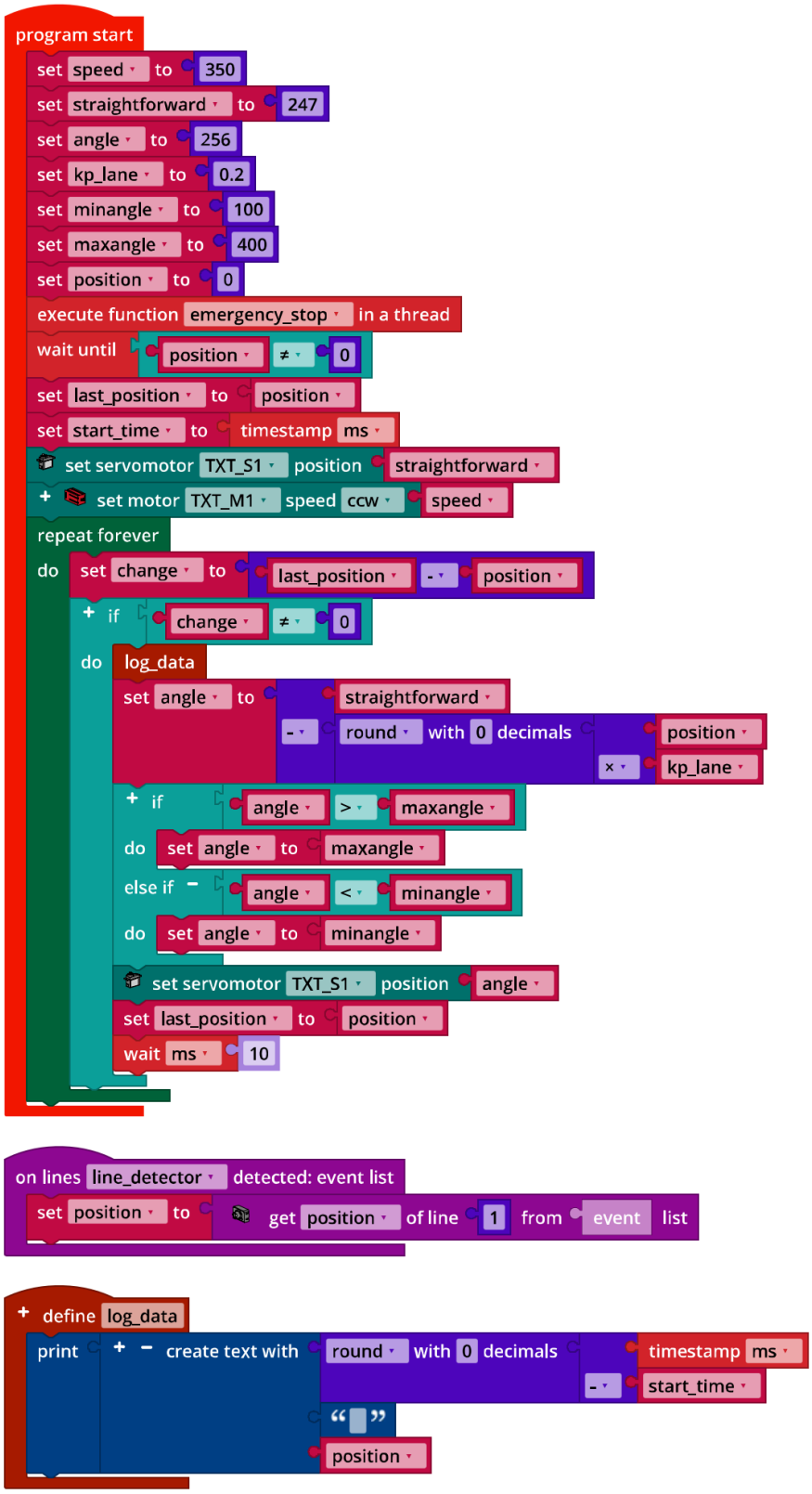


1a. Bucle de control del asistente de mantenimiento de carril:



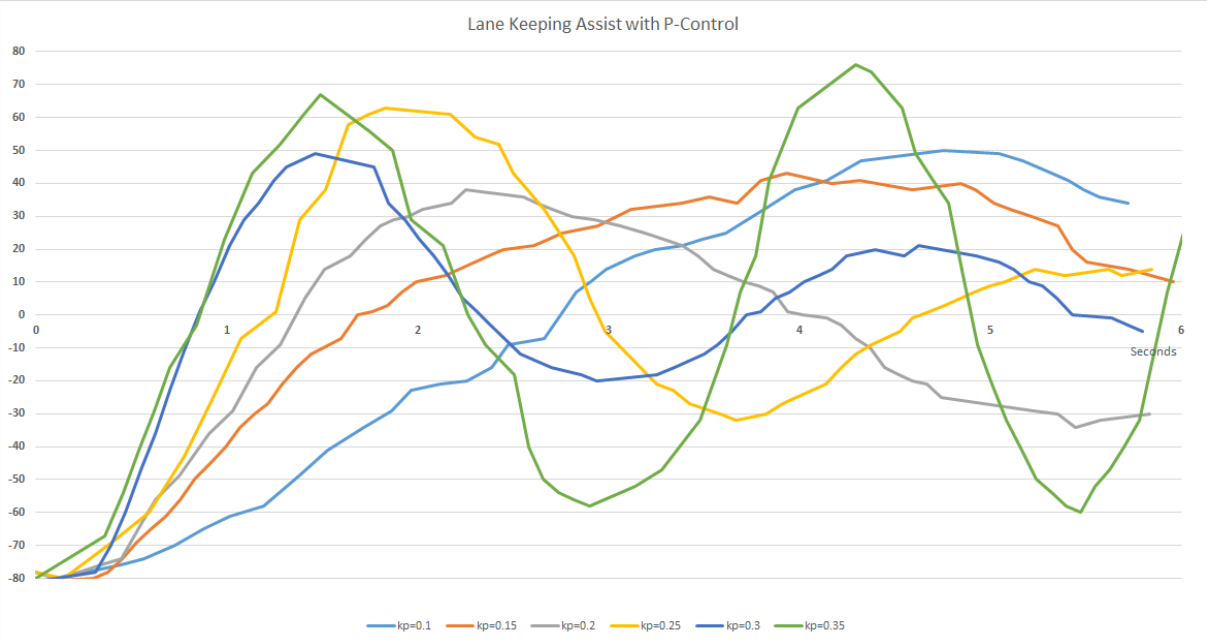
Regelkreis_Spurhalteassistent.drawio

1b. Programa (ejemplo) del asistente de mantenimiento de carril con controlador P:



Lane_Keeping_Assist_P-Control.ft

1c. Resultados de medición del controlador P a una velocidad de 350 (con $k_p \in \{0.1, 0.15, 0.2, 0.25, 0.3, 0.35\}$):



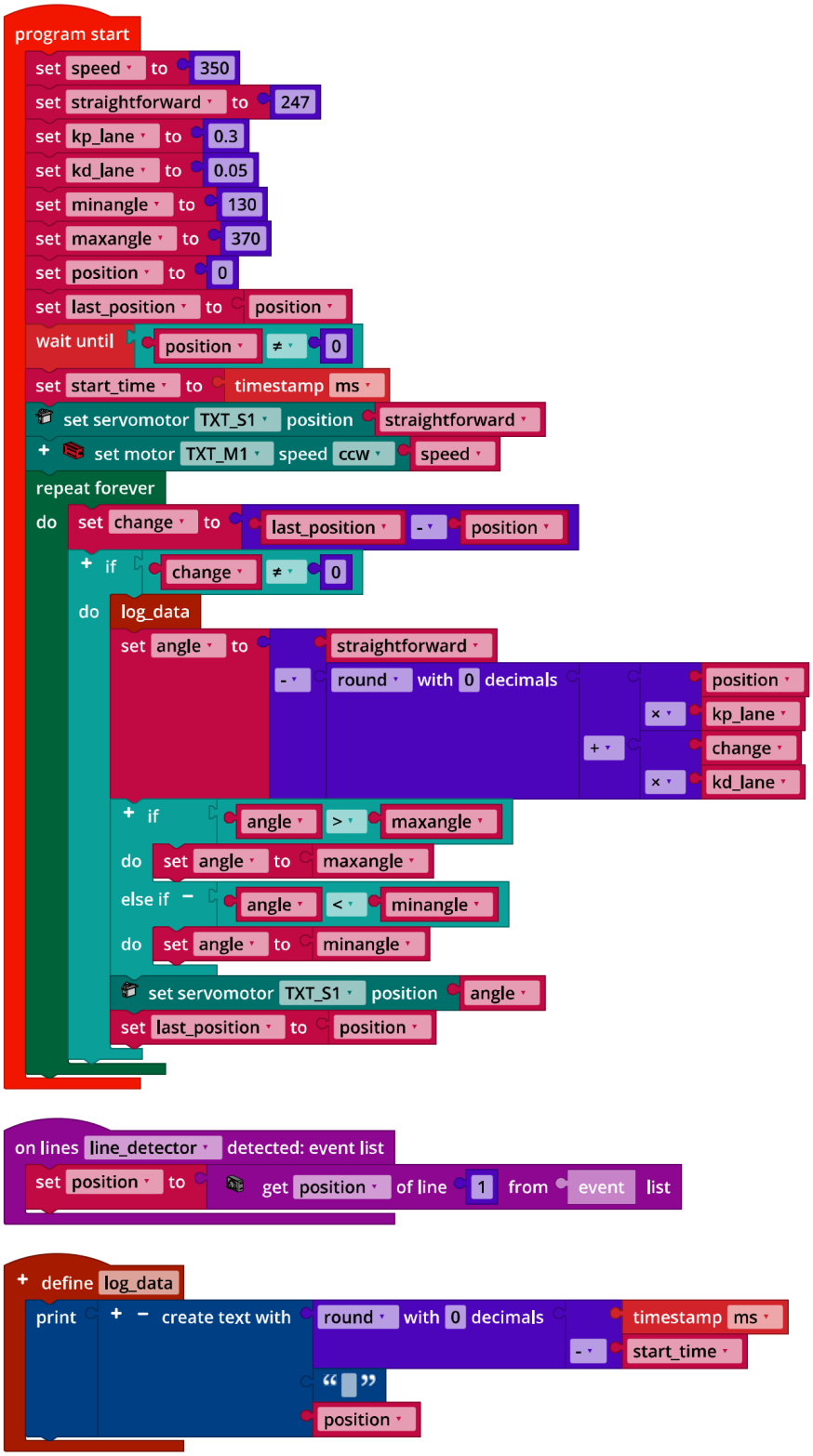
Lane_Keeping_Assist_P-Control_Results.jpg

La frecuencia máxima de fotogramas es de 15 fps (*frames per second*), es decir, la detección de líneas puede realizar una nueva determinación de posición como máximo cada 66,7 ms aproximadamente.

A un valor de $k_p = 0.35$ comienza a oscilar el controlador el deslizador en el ejemplo de solución. Con $k_p = 0.3$ el controlador oscila muy rápidamente con amplitud decreciente. En función del modelo (fricción, potencia del motor, ...) y del programa, los resultados de las mediciones de las alumnas y los alumnos pueden variar.

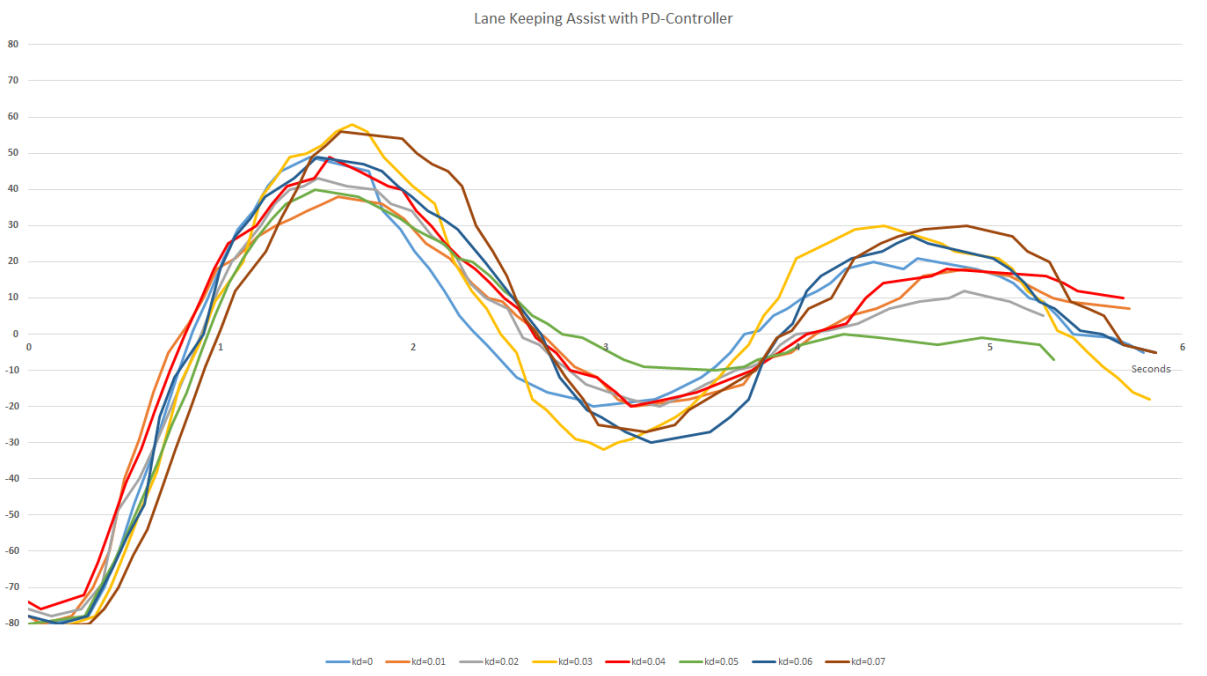
2. Asistente de mantenimiento de carril con controlador PD

2a. Programa (ejemplo):



Lane_Keeping_Assist_PD-Control.ft

2b. Resultados de medición del controlador PD a una velocidad de 350 y con $k_p = 0.3$ (para $k_d \in \{0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07\}$):



Lane_Keeping_Assist_PD-Control_Results.jpg

La mayor amortiguación se consigue en el programa de ejemplo con un factor diferencial $k_d = 0.05$ (línea verde en la figura). Aquí también pueden variar los resultados de las mediciones de las alumnas y los alumnos; sin embargo, en términos cualitativos, las mediciones deberían ser muy similares.

Anexos

Tarea n.º 2: Asistente de frenado, control de velocidad constante y asistente de mantenimiento de carril

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.
- Carril con marcas en una hoja adjunta (o el archivo del carril impreso).
- Obstáculo (por ejemplo, un libro o una caja de cartón)

Más información

[1] Jim Meininghaus: *Die Geschichte des Tempomaten. Wie ein Blinder das Autofahren veränderte*. 03/03/2014, motor-talk.de.

[2] Thomas Paulsen: *Autonomes Fahren: Die 5 Stufen zum selbstfahrenden Auto*. 07/11/2018, adac.de.

[3] Editor de diagramas en línea para crear diagramas de transición de estados (formato drawio): https://www.diagrammeditor.de/

Nombre: _____ Clase: ___ Fecha: _____

Tarea n.º 3

Control automático de las luces

En el tráfico vial, los sistemas de asistencia al conductor de nuestros vehículos pueden provocar accidentes – el resto del tráfico no es advertido ni por un intermitente ni por una luz de freno. Por ello ahora equipamos el vehículo con funciones de iluminación.

Tarea de construcción

Conecta las dos luces delanteras (luces de cruce) en paralelo a la salida O5 y la luz de marcha atrás (blanca) a la salida O4. La luz de freno (roja) se conecta a O3. El intermitente izquierdo se conecta a O7 y el derecho a O8.

Tareas de programación

A tu programa de control con sistemas de asistencia al conductor ahora se añaden las funciones de iluminación automática.

1. Luz de marcha atrás

La luz de marcha atrás debe encenderse cuando el motor gira hacia atrás. Añade un hilo correspondiente al programa de control.

2. Luz de freno

La luz de freno debe encenderse cuando el motor disminuye la velocidad.

2a. Añade un hilo correspondiente al programa.

2b. La luz de freno debe permanecer encendida durante 0,5 segundos para que pueda ser vista por el vehículo que viene detrás incluso ante una ligera desaceleración.

2c. Amplía la función de la luz de freno teniendo en cuenta la magnitud de la desaceleración: Cuanto mayor sea la acción de frenado, más brillante deberá ser la luz de freno.

3. Intermitente

Los dos intermitentes deben activarse automáticamente cuando la dirección supera un ángulo determinado (aquí: palanca del servomotor en una posición inferior a 190 o

superior a 310). Observa: De acuerdo con el reglamento de tráfico, un intermitente debe parpadear con una frecuencia de 1-2 Hz.

La información sobre si el intermitente está activado debe intercambiarse entre el programa principal y el hilo a través de un semáforo (aquí: una variable de estado compartida).

Tareas experimentales

1. Luces de emergencia

En caso de una «parada de emergencia» o de una frenada a causa de un obstáculo, debe activarse una luz de emergencia.

Amplía el hilo del intermitente como corresponde.

2. Luz de cruce

Las luces delanteras y traseras deben encenderse automáticamente cuando oscurece (véase también la tarea n.º 6 del Robotics TXT 4.0 Base Set).

Para ello, coloca el fototransistor en tu vehículo y conéctalo a la entrada I7.

Programa el control automático de las luces como otro hilo.

Dado que la luz trasera se utiliza también como luz de freno, solo puedes activarla con la mitad del brillo. Ajusta la luz de freno automática para que la luz trasera brille más al frenar (véase la tarea de programación n.º 2c más arriba).

Anexos

Tarea n.º 3: Control automático de las luces

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.

Más información

[1] Wikipedia: *Concurrencia*.

Tarea n.º 3: Control automático de las luces

En esta tarea, el vehículo autónomo está equipado con una luz de cruce de activación automática, una luz de freno, una luz de marcha atrás, así como intermitentes y una luz de emergencia.

Tema

Iluminación del vehículo controlada automáticamente por procesos secundarios (paralelos) dependientes del tiempo (intermitentes, luces de emergencia) y circuitos de retardo (luces de freno).

Objetivo de aprendizaje

- Uso de hilos para programar procesos secundarios (casi simultáneos)
- Comunicación entre hilos mediante semáforo
- Dependencia temporal de los procesos (intervalo de tiempo, retrasos)

Tiempo necesario

Deben conectarse los seis LED del vehículo autónomo. Para ello, las alumnas y los alumnos necesitan alrededor de 15-20 minutos.

Para la resolución de la tarea de programación, las alumnas y los alumnos necesitan una o dos clases (45-90 minutos).

Anexos
Tarea n.º 3: Control automático de las luces

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.

Más información

[1] Wikipedia: *Concurrencia*.

Nombre: _____ Clase: ___ Fecha: _____

Hoja de soluciones de la tarea temática n.º 3

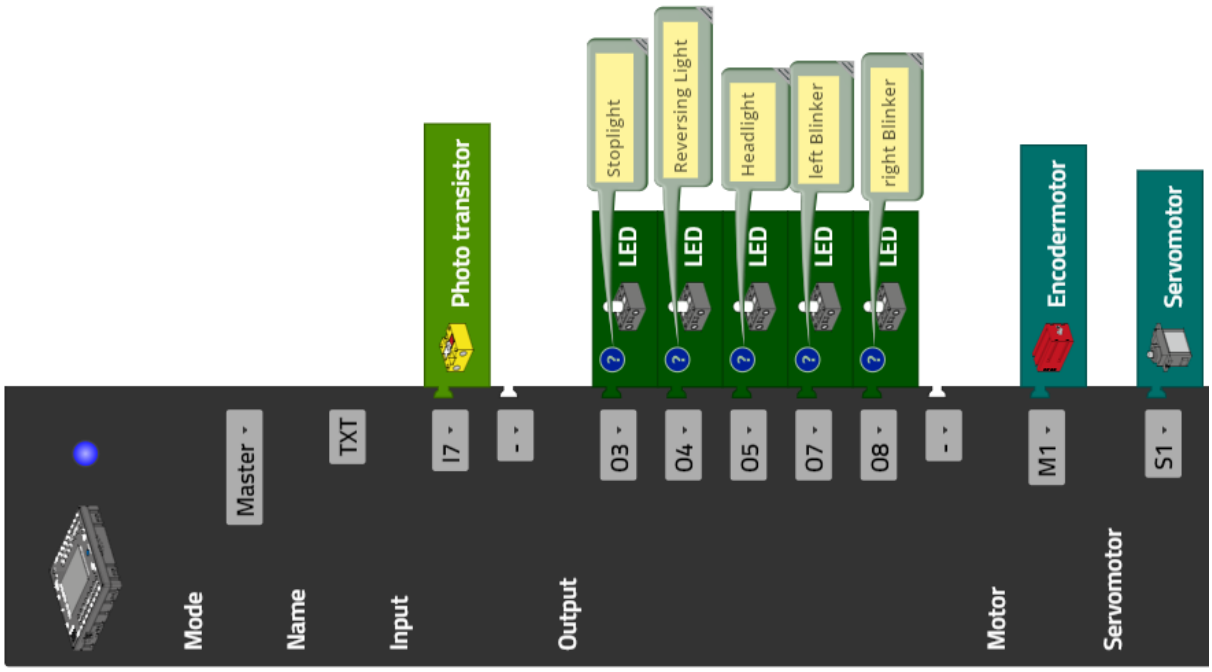
Control automático de las luces

Cada uno de los hilos consulta un «disparador»: la condición desencadenante, como la velocidad o la dirección de giro del motor o el ángulo de giro del servomotor. Un buen ejercicio para el empleo de procesos concurrentes: El control de la iluminación está completamente desacoplado del control del motor, lo que hace que el programa sea más claro y menos propenso a errores de programación.

Tarea de construcción

En las siguientes tareas solo se necesitan el LED, el fototransistor, el motor y el servomotor.

Configuración de los sensores:



Tareas de programación

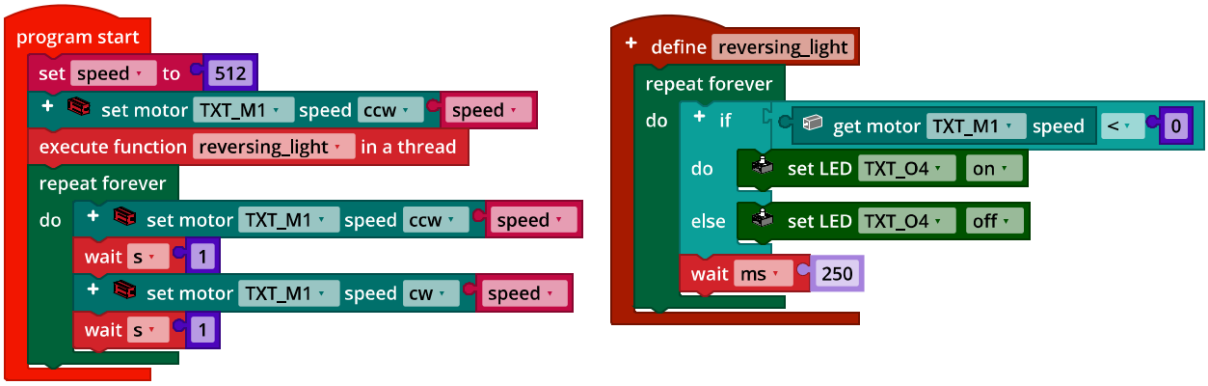
La condición determinante para activar los LED se consulta directamente o a través de variables (como semáforos) en el hilo respectivo. En los siguientes ejemplos de programas, el programa principal contiene sencillas rutinas de prueba con las que se puede comprobar el funcionamiento del control automático de las luces.

El programa «*Automatic_Ligthing.ft*» reúne todos los hilos para la iluminación.

1. Luz de marcha atrás

En el caso de la luz de marcha atrás, es necesario evaluar la dirección de giro del motor.

Programa (ejemplo) de la luz de marcha atrás con programa de prueba:



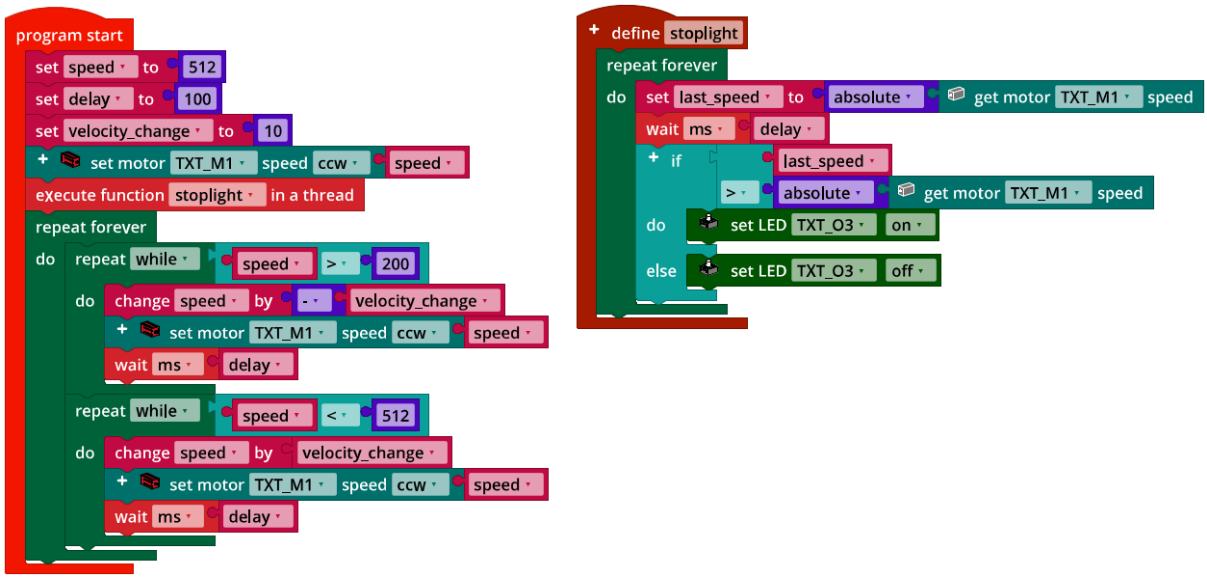
Reversing_Light.ft

2. Luz de freno

Para que la luz de freno se ilumine cuando desacelera el motor es necesario medir las velocidades sucesivas del motor.

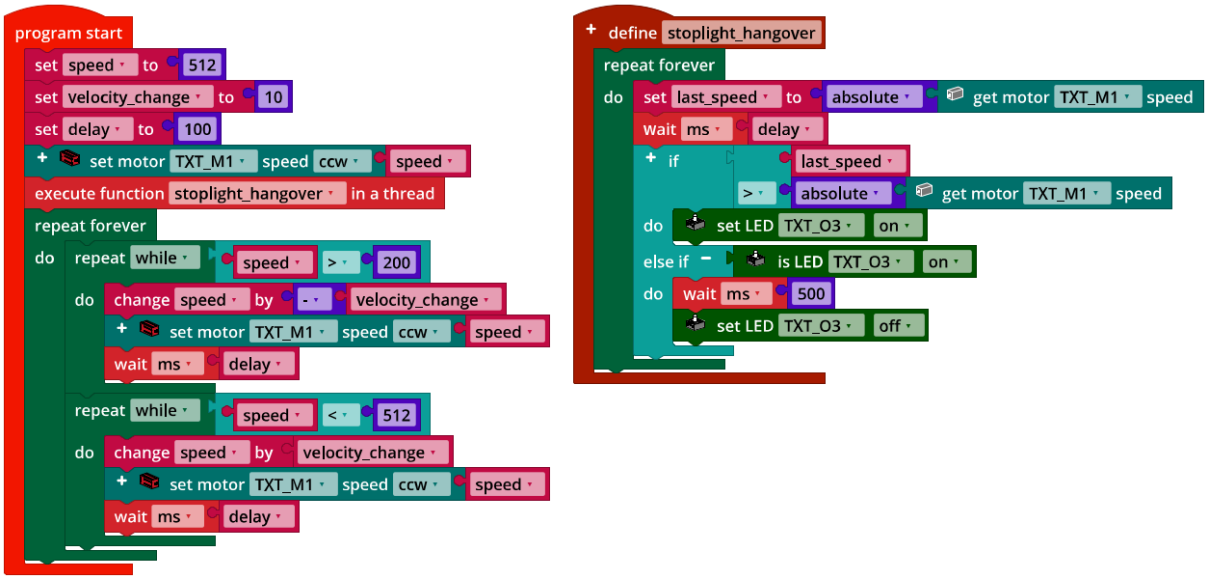
Es importante observar: Para que la luz de freno se ilumine correctamente durante la marcha atrás es necesario comparar las velocidades en la trayectoria para detectar la desaceleración.

2a. Programa (ejemplo) de la luz de freno con programa de prueba:



Stoplight.ft

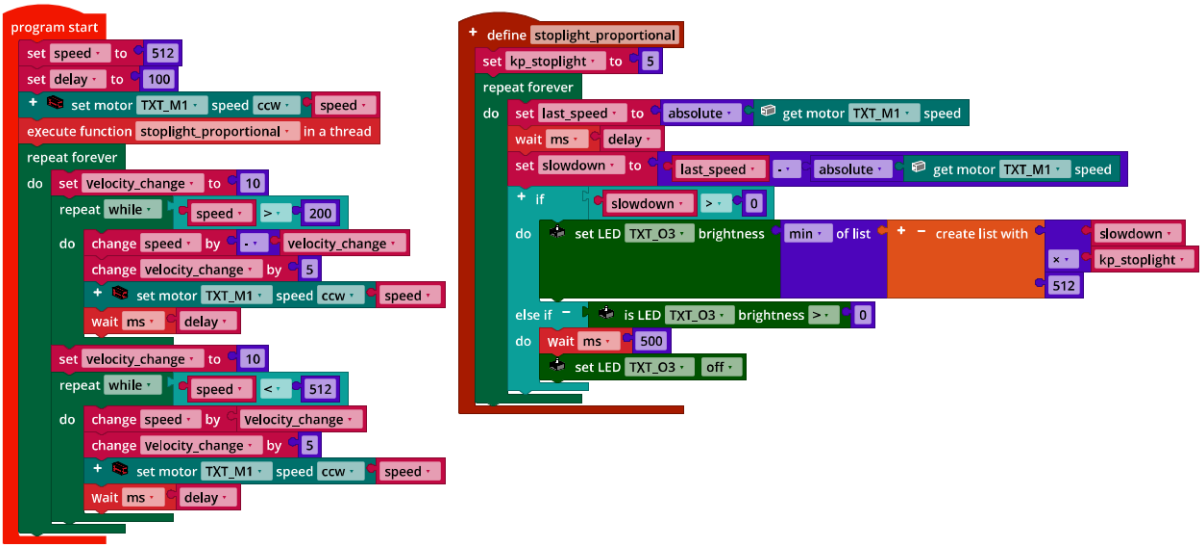
2b. Programa (ejemplo) de la luz de freno con función de apagado retardado y programa de prueba:



Stoplight_hangover.ft

2c. La duración de la medición en el hilo «stoplight» tiene una influencia considerable en la magnitud del cambio de velocidad; es necesario experimentar con este valor para representar una reacción adecuada en un comportamiento de frenado realista. Con un factor de proporcionalidad (k_p) puede determinarse la influencia del cambio de velocidad en el brillo.

Programa (ejemplo) de la luz de freno con brillo proporcional y programa de prueba:



Stoplight_proportional.ft

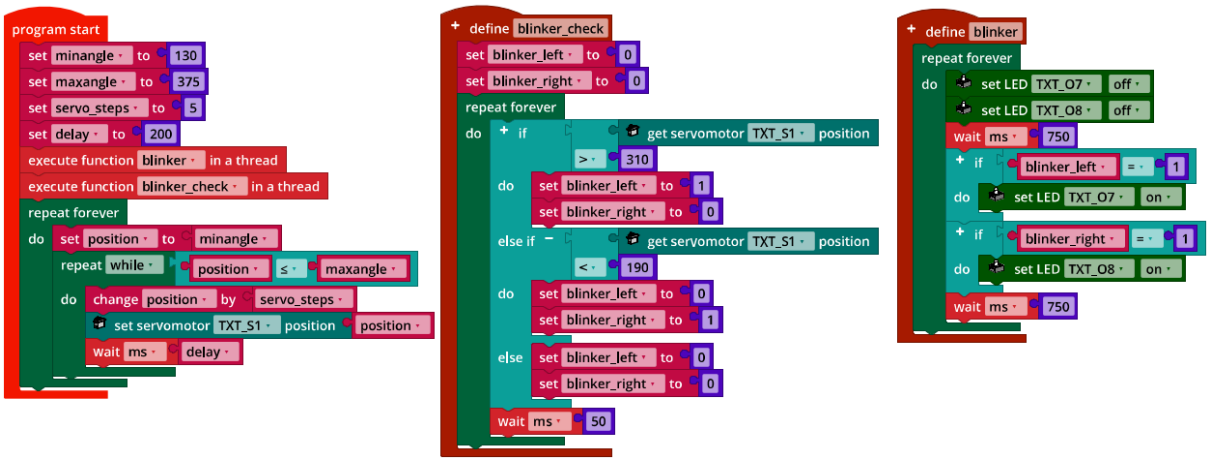
3. Intermitente

El principio de funcionamiento de un intermitente ya se conoce de la tarea n.º 3 del Robotics TXT 4.0 Base Set .

Para el intermitente se requieren varias variables de estado: una para cada intermitente, que indica si está activado el intermitente derecho o el izquierdo en función del ángulo de dirección (es decir, la posición del servomotor) y un estado que distingue si la luz intermitente (en caso de que el intermitente esté activado) está encendida o apagada.

El siguiente ejemplo de programa resuelve la tarea con dos hilos. Ventaja de este enfoque: Puede añadirse fácilmente una luz de emergencia.

Programa (ejemplo) de intermitente con programa de prueba:



Blinker.ft

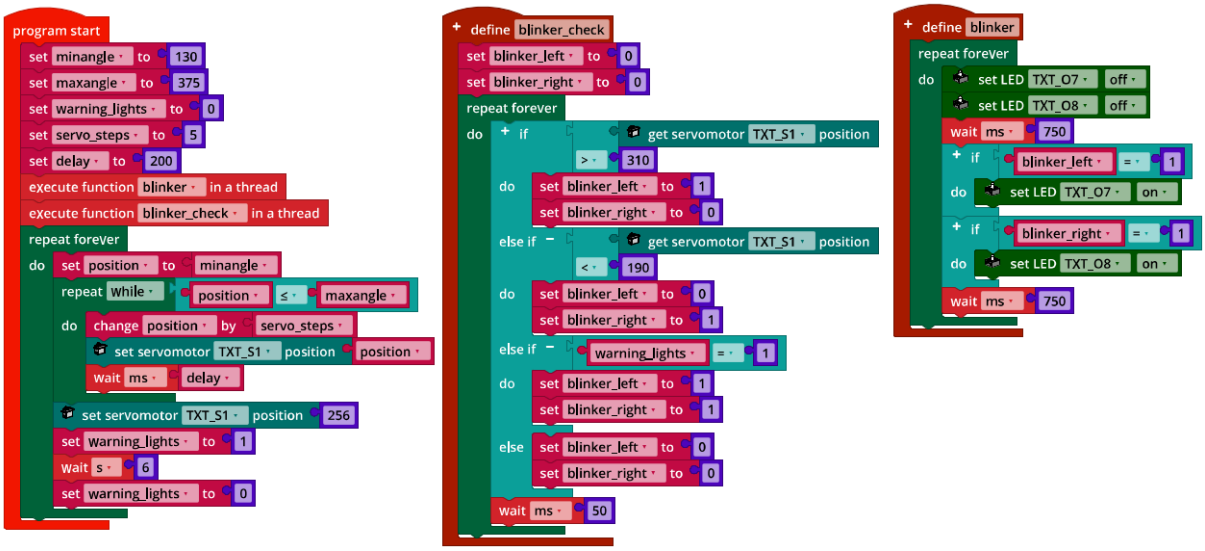
Las variables de estado «blinker_left» y «blinker_right» sirven como semáforos entre el hilo «blinker_check» y «blinker».

Tareas experimentales

1. Luces de emergencia

El estado de la luz de emergencia puede ser señalado al hilo a través de una variable como semáforo. Entonces es suficiente con ampliar el hilo de prueba de los intermitentes: Cuando se encienden las luces de emergencia se activan los dos intermitentes y se encienden y apagan en el hilo de los intermitentes.

Programa (ejemplo) de las luces de emergencia con programa de prueba:

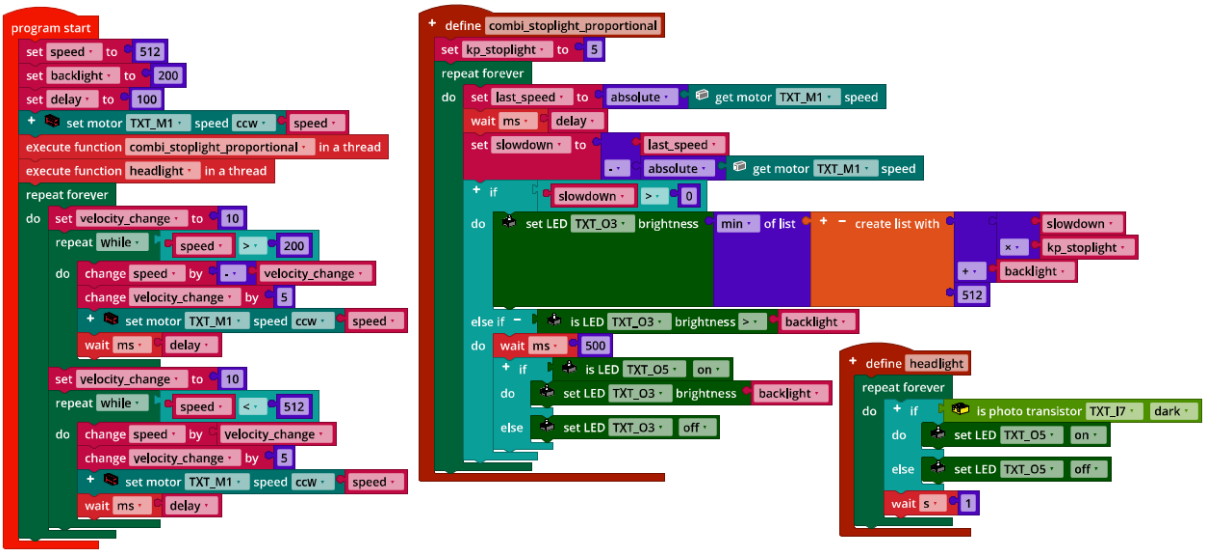


Blinker_with_Warning_Lights.ft

2. Luz de cruce

El brillo de la luz trasera solo puede modificarse si el LED no se utiliza al mismo tiempo como luz de freno. Si en el hilo para la luz de freno se utiliza el estado de las luces delanteras como indicador de activación de la luz de cruce, entonces el control de la luz trasera y de freno combinadas puede llevarse a cabo en el mismo hilo.

Programa (ejemplo) de luz de cruce (programa de prueba):



Headlight.ft

Anexos

Tarea n.º 3: Control automático de las luces

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.

Más información

[1] Wikipedia: *Concurrencia*.

Nombre: _____ Clase: ___ Fecha: _____

Tarea n.º 4

Asistente de aparcamiento

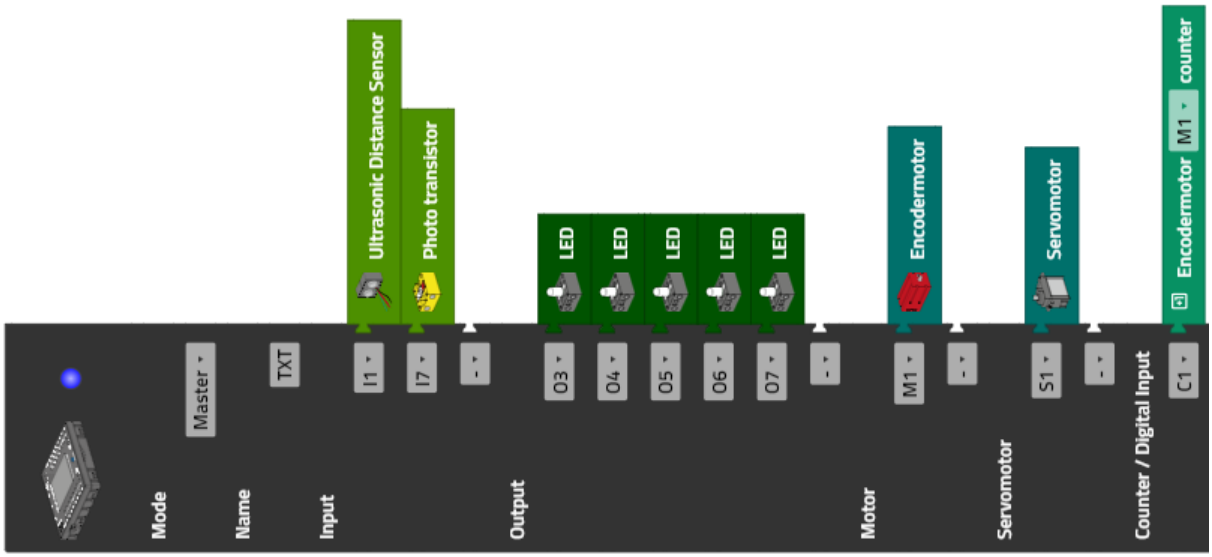
Ahora equipamos el vehículo con otro sistema de asistencia al conductor importante: un asistente de aparcamiento. A través de él debe encontrar por sí mismo una plaza de aparcamiento lo suficientemente grande y aparcar en ella en dos movimientos.

Tarea de construcción

Monta el sensor ultrasónico en tu vehículo de modo que pueda medir la distancia a la que se encuentra el vehículo con respecto a objetos situados en el borde derecho de la carretera.

Tareas de programación

Configuración de los sensores:



1. Encontrar plazas de aparcamiento

Ahora debes desarrollar un programa que se dirija a una distancia de aproximadamente 2 cm del límite derecho del carril y que utilice el sensor ultrasónico para encontrar una plaza de aparcamiento lo suficientemente grande. Puedes delimitar las plazas de aparcamiento con objetos (por ejemplo, cajas de cartón). Al

final del primer espacio lo suficientemente grande, el vehículo debe detenerse en el punto donde puede comenzar la maniobra de aparcamiento.

Para ello, primero determina manualmente el tamaño mínimo (es decir, la longitud y la anchura) de una plaza de aparcamiento para que su vehículo pueda aparcar en marcha atrás. Debes tener en cuenta la dispersión de la señal ultrasónica: ¿Cuándo detecta el sensor el comienzo y el final de una plaza de aparcamiento si estos están delimitados por objetos situados a una distancia de aproximadamente 2 cm a la derecha de la línea de delimitación del carril?

Finalmente, determina a qué distancia del final de la plaza de aparcamiento debe detenerse el vehículo para poder iniciar la maniobra de aparcamiento.

1a. Ilustra la maniobra de aparcamiento y tus mediciones con un dibujo.

1b. Diseña un diagrama de transición de estados.

1c. Programa la búsqueda de la plaza de aparcamiento. Utiliza para ello el asistente de mantenimiento de carril de la tarea n.º 2.

Prueba el programa en tramo de carril de la hoja adjunta con obstáculos ubicados a diferentes distancias a la derecha del límite del carril.

2. Maniobra de aparcamiento

En el segundo paso, después de encontrar una plaza de aparcamiento, el vehículo debe aparcar en marcha atrás en dos movimientos.

2a. Primero vuelve a determinar manualmente (prueba de interfaz) cuántos impulsos debe retroceder el vehículo primero con una maniobra hacia la derecha y a continuación con una maniobra hacia la izquierda.

2b. ¿Cuántos impulsos se necesitan para que el vehículo se detenga en el centro (de la parte necesaria para aparcar) de la plaza de aparcamiento?

2c. Programa la maniobra de aparcamiento y pruébala en el tramo de carril.

Tareas experimentales

1. Asistente de aparcamiento

Ahora amplía tu asistente de aparcamiento con las funciones desarrolladas en la tarea n.º 3 para que cumpla todos los requisitos del reglamento de tráfico al aparcar (luz de freno, luz de marcha atrás, intermitentes, luz de cruce). Añade al programa una «señal de éxito» que suene cuando se haya completado la maniobra de aparcamiento y haz que parpadeen las luces de emergencia cinco veces.

2. Cálculo de la maniobra de aparcamiento

La maniobra de aparcamiento también puede calcularse: El centro del diferencial describe dos segmentos del círculo cuyos radios pueden derivarse del ángulo de giro de la dirección.

2a. Dibuja la fase de la maniobra de aparcamiento en la que el vehículo se desplaza hacia atrás.

2b. Calcula los impulsos necesarios a partir de los ángulos y algunos ajustes sobre las distancias correctas a la línea lateral.

Compara el resultado con los valores determinados a través de la experimentación en la tarea de programación.

Anexos

Asistente de aparcamiento

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.
- Trayecto de prueba

Más información

[1] VDA: *Automatisierung: Von Fahrerassistenzsystemen zum automatisierten Fahren*. Septiembre de 2015.

[2] Nina Tetzlaff: *Autonome Straßenfahrzeuge*. Erfinderaktivitäten 2018/2019, DPMA, Págs. 62-81.

[3] VW: *Park Assist Steering 2.0*. Design and function. Service Training, 2021.

Tarea n.º 4: Asistente de aparcamiento

El vehículo está equipado con un asistente de aparcamiento, mediante el cual puede buscarse una plaza de aparcamiento lo suficientemente grande y aparcar marcha atrás en dos movimientos.

Tema

Programación de un sistema de asistencia al conductor (asistente de aparcamiento).

Objetivo de aprendizaje

- Medición de un objeto a partir del movimiento, teniendo en cuenta la dispersión ultrasónica
- «Divide y vencerás»: Procedimiento para reducir la complejidad mediante la fragmentación de un problema en problemas parciales independientes
- Cálculo trigonométrico de una secuencia de movimiento

Tiempo necesario

En el vehículo autónomo solo debe fijarse el sensor ultrasónico en su lateral.

Para el desarrollo del programa para resolver las tareas, las alumnas y los alumnos necesitan entre dos y cuatro clases (90-180 minutos).

Anexos
Asistente de aparcamiento

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.
- Trayecto de prueba

Más información

[1] VDA: *Automatisierung: Von Fahrerassistenzsystemen zum automatisierten Fahren*. Septiembre de 2015.

[2] Nina Tetzlaff: *Autonome Straßenfahrzeuge*. Erfinderaktivitäten 2018/2019, DPMA, Págs. 62-81.

[3] VW: *Park Assist Steering 2.0*. Design and function. Service Training, 2021.

Nombre: _____ Clase: ___ Fecha: _____

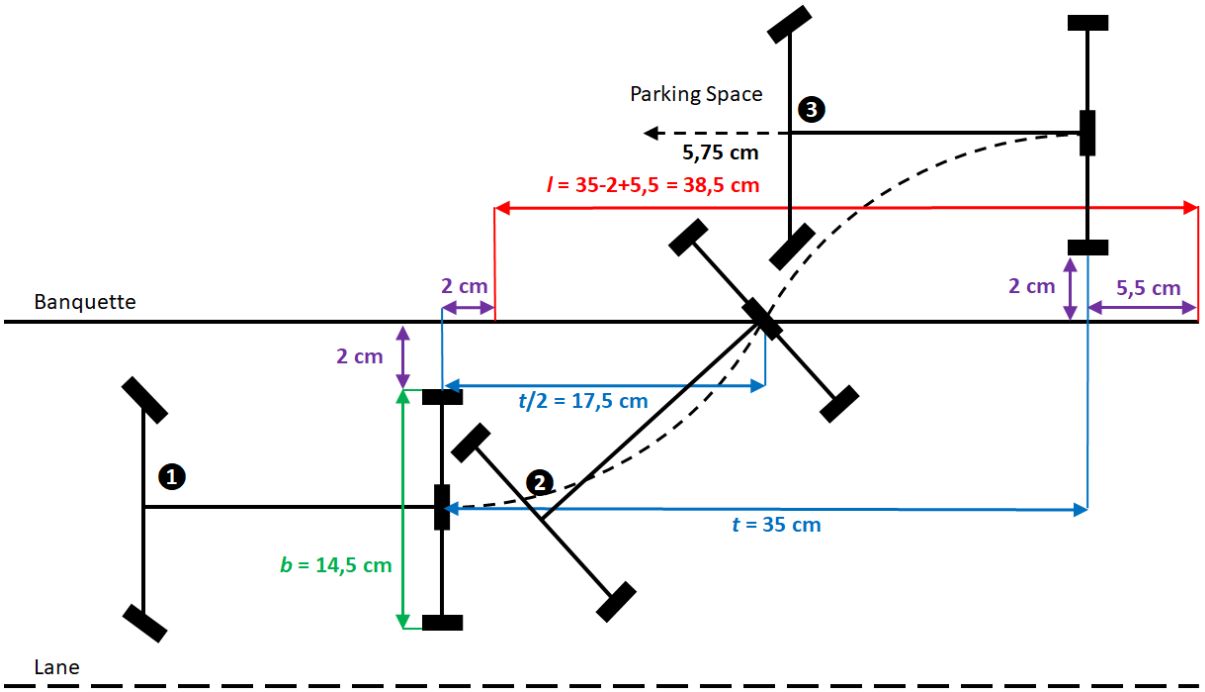
Hoja de soluciones de la tarea temática n.º 4

Asistente de aparcamiento

Tareas de programación

1. Encontrar plazas de aparcamiento

La maniobra de aparcamiento puede reproducirse fácilmente a mano con la palanca del servomotor desmontada. La forma más sencilla de medir la longitud de la plaza de aparcamiento necesaria es con la ayuda de las correspondientes marcas de lápiz en el carril (véase la figura).



Parallel_Parking_Model.jpg

Anchura de la plaza de aparcamiento: La plaza de aparcamiento debe ser al menos tan ancha como el vehículo, es decir b = 14,5 cm, más 2 cm hasta el límite del carril.

La plaza debe ser 1,5 cm más amplia porque los neumáticos exceden la anchura del vehículo al maniobrar.

Si el sensor ultrasónico se monta de modo que mida desde el borde exterior del vehículo, añade otros 2 cm hasta la línea lateral y su anchura (2 cm).

Así, se cuenta con un espacio lo suficientemente amplio a la derecha del vehículo cuando el sensor mide **al menos 22 cm** hasta el objeto más cercano.

Longitud de la plaza de aparcamiento: Al aparcar en dos movimientos, el centro del diferencial debe retroceder primero unos 17,5 cm con el máximo ángulo de dirección al maniobrar hacia la derecha antes de cruzar el centro de la línea lateral y, a continuación, otros 17,5 cm con el máximo ángulo de dirección al maniobrar hacia la izquierda. A esta distancia de 35 cm debe añadirse longitud de la parte posterior del vehículo (5,5 cm medidos desde el centro del diferencial hasta el extremo del vehículo) y restar la «distancia inicial» de 2 cm (entre la posición del diferencial al iniciar la maniobra de aparcamiento y el punto en el que la rueda delantera izquierda cruza el arcén). De este modo, la longitud mínima requerida de la plaza de aparcamiento asciende a

$$l = 38{,}5\ \text{cm}$$

A partir de la longitud mínima de la plaza de aparcamiento l puede determinarse el número de impulsos i necesarios para esta distancia a partir de la siguiente fórmula (véase la tarea n.º 1):

$$i = l \cdot \frac{1}{20{,}5} \cdot \frac{13}{7} \cdot 63{,}9\ \frac{1}{\text{cm}} \approx l \cdot\ 5{,}789\ \frac{1}{\text{cm}}$$

Si observamos el factor de conversión adaptado de la prueba en la tarea n.º 1, la conversión se corrige a

$$i = l \cdot \frac{100}{0{,}0017} \approx l \cdot\ 5{,}882\ \frac{1}{\text{cm}}$$

Así, la longitud de la plaza de aparcamiento buscada requiere de $l = 38{,}5\ \text{cm}$ o sea $\boldsymbol{i \approx 227}$ **impulsos**.

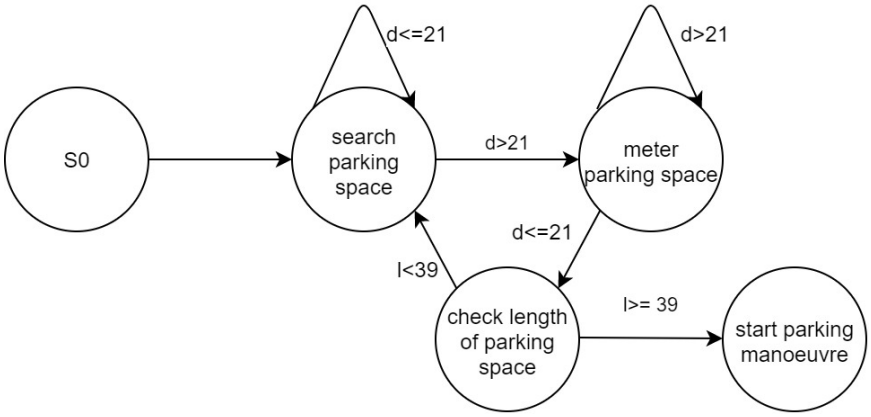
Dispersión de la señal ultrasónica: Si hay obstáculos a una distancia de 4 cm a la derecha del límite del carril, la distancia al sensor ultrasónico es de 8 cm. Debido al ángulo de dispersión del sensor ultrasónico, la plaza de aparcamiento se detecta solo cuando el centro del sensor ultrasónico «se adentra» 3 cm en la plaza de aparcamiento. El final de la plaza de aparcamiento se detecta 3 cm antes de que el centro del sensor alcance el borde exterior del obstáculo. Por lo tanto, si en una distancia de 32,5 cm (o $\boldsymbol{i' \approx 191}$ **impulsos**) el sensor mide una anchura de al menos 22 cm significa que la plaza de aparcamiento es lo suficientemente larga.

Para medir la longitud de la plaza de aparcamiento es irrelevante en qué sitio se fije el sensor ultrasónico en el lateral del vehículo. Para iniciar la maniobra de aparcamiento, el vehículo debe detenerse de forma que el eje trasero (o el diferencial) se encuentre exactamente a 2 cm por detrás del final de la plaza de aparcamiento (es decir, el borde más externo del obstáculo). Para ello, después de que el sensor ultrasónico haya detectado el final de la plaza de aparcamiento, debe continuar

- otros 3 cm (por los que se detecta el final de la plaza de aparcamiento demasiado pronto)
- más los 2 cm anteriores
- más la distancia desde el centro del sensor ultrasónico al eje trasero.

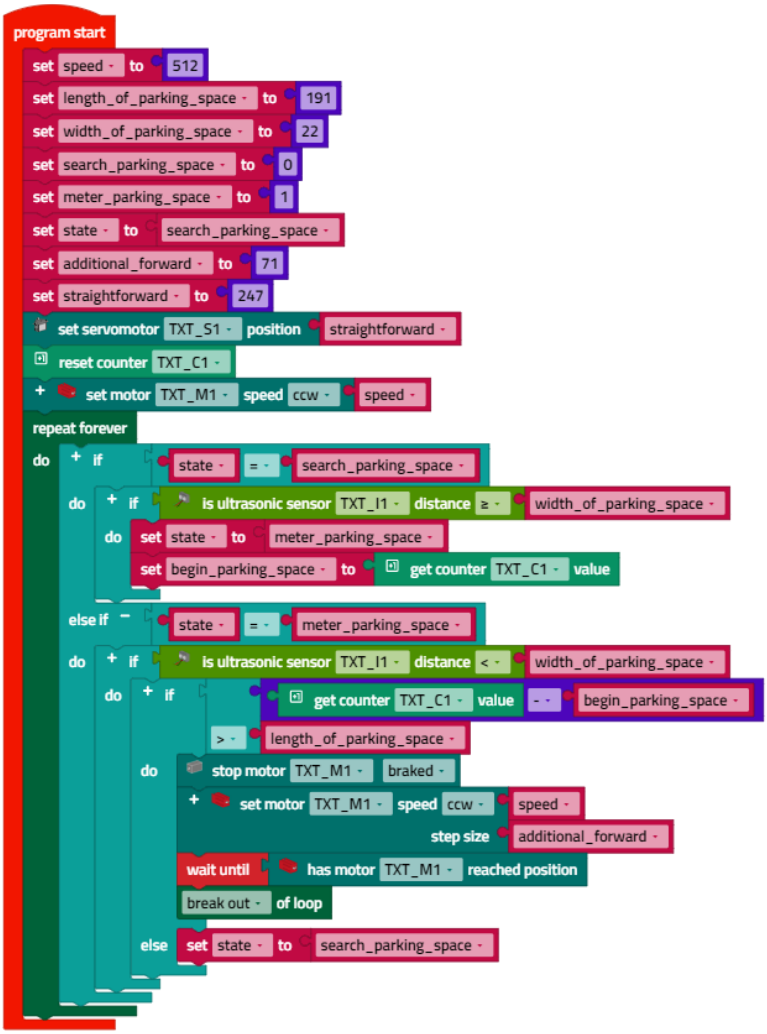
. Si el sensor ultrasónico está ubicado, por ejemplo, 7 cm delante del eje trasero, el vehículo debe desplazarse 12 cm más (o **71 impulsos**) después de detectar el final de la plaza de aparcamiento.

1b. Diagrama de transición de estados:



State-Transistion_Diagram_Find_Parking_Space.drawio

1c. Programa (ejemplo) de búsqueda de la plaza de aparcamiento:



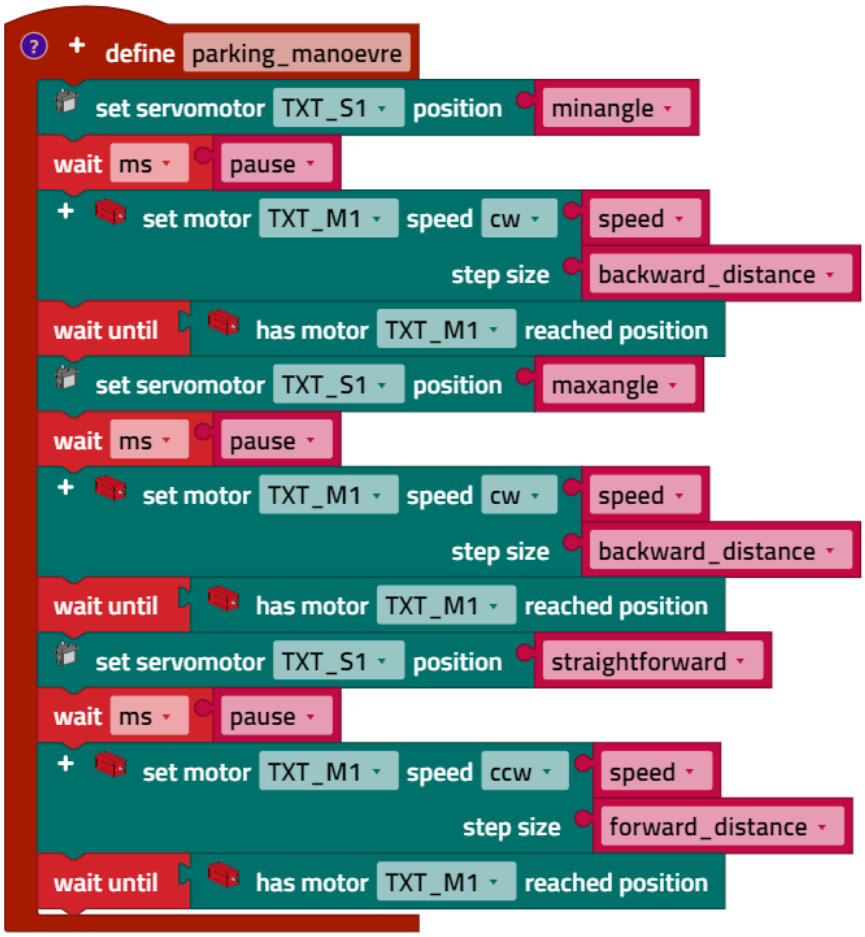
Find_Parking_Space.ft

2. Maniobra de aparcamiento

2a. La forma más sencilla de determinar los impulsos por realizar es a través de la prueba de interfaz: Se aparca el vehículo de forma manual; los impulsos emitidos por el codificador se leen y se anotan en la prueba de interfaz. Para retroceder se requieren – con la dirección orientada hacia la derecha o hacia la izquierda – **125 impulsos respectivamente**.

2b. Para que el vehículo finalmente se ubique en el centro de la zona de aparcamiento requerida debe avanzar 5,75 cm después retroceder; esto equivale aproximadamente a **34 impulsos**.

2c. Programa (ejemplo) de maniobra de aparcamiento:



Parking_Manoevre.ft

Las variables «backward_distance» y «forward_distance» contienen los impulsos a realizar (125 y 34). «minangle» y «maxangle» son los máximos ángulos de dirección posibles del servomotor en el soporte (véase la tarea n.º 2); son de aproximadamente 130 y 375. Para la pausa breve después de una maniobra («pause»), 250 ms es una buena opción.

Tareas experimentales

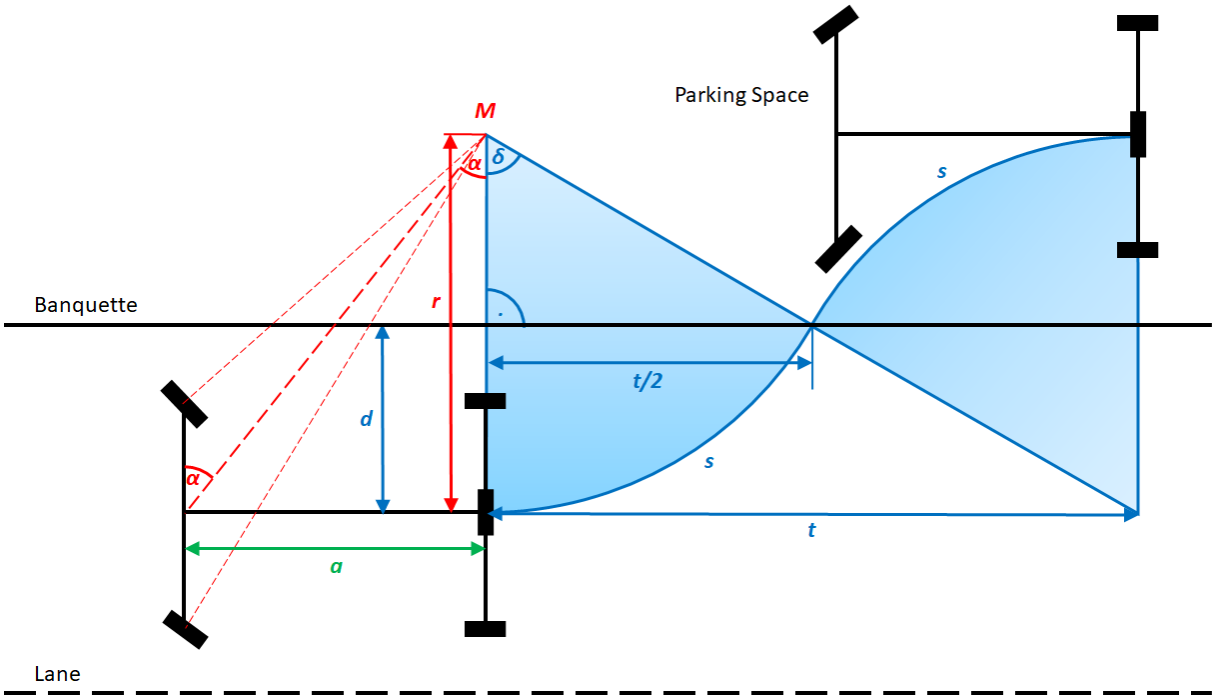
1. Asistente de aparcamiento

Una solución de esta tarea se resume en el programa *Autonomous_Parking.ft*.

El intermitente no debe activarse automáticamente; de lo contrario, se cambiará al intermitente izquierdo en medio de la maniobra de aparcamiento.

2. Cálculo de la maniobra de aparcamiento

En la maniobra de aparcamiento, el centro del diferencial se desplaza exactamente a lo largo de dos segmentos circulares de longitud s. El centro M del primer círculo corresponde al centro del círculo de giro en nuestra mangueta de dirección: Las extensiones de los dos muñones del eje delantero de la dirección y del eje de la rueda trasera se encuentran exactamente en este punto.



Mathematical_Model_Parallel_Parking.jpg

Ahora calculamos la longitud de este segmento circular s. Esto nos indica cuántos impulsos del codificador necesitamos para recorrer este segmento. Se aplica: s se comporta con respecto a la circunferencia ($= 2\pi \cdot r$) como δ con respecto a 360°, es decir:

$$s = \frac{\pi \cdot \delta \cdot r}{180°}$$

El ángulo δ se expresa a través del radio r, ya que en el triángulo rectángulo formado por M y los dos puntos de intersección de las líneas del segmento circular con la línea lateral se aplica:

$$\cos\delta = \frac{(r-d)}{r}$$

Para la distancia d desde el centro del eje hasta (el centro de) la línea lateral se presupone que el vehículo se encuentra aproximadamente a 2 cm a la izquierda del límite del carril. Así, con una anchura del vehículo de b = 14,5 cm y nuestra línea lateral de 2 cm se aplica:

$$d = 10{,}25\ \text{cm}$$

El radio r del círculo de giro lo calculamos a partir del triángulo formado por M y los dos centros de los ejes delantero y trasero del vehículo. La distancia entre los ejes la identificamos como a:

$$\tan\alpha = \frac{a}{r}\text{, o sea: } r = \frac{a}{\tan\alpha}$$

El ángulo α equivale al ángulo de giro (máximo) de nuestra palanca del servomotor. En nuestro vehículo, el ángulo de giro de la dirección es de aproximadamente 38°. Podemos medir la distancia de los ejes: $a = 15{,}5\ \text{cm}$.

Finalmente, obtenemos (con $\alpha = 38°$):

$$s = \frac{\pi}{180°} \cdot \arccos\left(\frac{a - d\cdot\tan\alpha}{a}\right) \cdot \frac{a}{\tan\alpha} \approx 21{,}15\ \text{cm}$$

Esto equivale aproximadamente a 122 impulsos.

Ahora también podemos calcular la longitud necesaria de la plaza de aparcamiento. Según el *teorema de Pitágoras*:

$$r^2 = \left(\frac{t}{2}\right)^2 + (r-d)^2$$

A continuación (con $\alpha = 38°$):

$$t = 2\cdot\sqrt{\left(\frac{a}{\tan\alpha}\right)^2 - \left(\frac{a}{\tan\alpha} - d\right)^2} \approx 34{,}73\ \text{cm}$$

A este valor se debe sumar la longitud de la parte trasera del vehículo (5,5 cm) y restar los 2 cm que se recorren más allá de la plaza de aparcamiento. Entonces, nuestra plaza de aparcamiento debería tener una longitud mínima de 38,23 cm - esto se corresponde con bastante precisión con el valor determinado a través de la experimentación.

Anexos

Asistente de aparcamiento

Material necesario

- Ordenador para el desarrollo de programas, localmente o a través de la interfaz web.
- Cable USB o conexión BLE o wifi para transferir el programa al TXT4.0.
- Trayecto de prueba

Más información

[1] VDA: *Automatisierung: Von Fahrerassistenzsystemen zum automatisierten Fahren*. Septiembre de 2015.

[2] Nina Tetzlaff: *Autonome Straßenfahrzeuge*. Erfinderaktivitäten 2018/2019, DPMA, Págs. 62-81.

[3] VW: *Park Assist Steering 2.0*. Design and function. Service Training, 2021.