Name: \_\_\_\_\_\_\_\_\_\_ Klasse: \_\_\_ Datum: \_\_\_\_\_\_\_\_

# Lösungsblatt Aufgabe 3

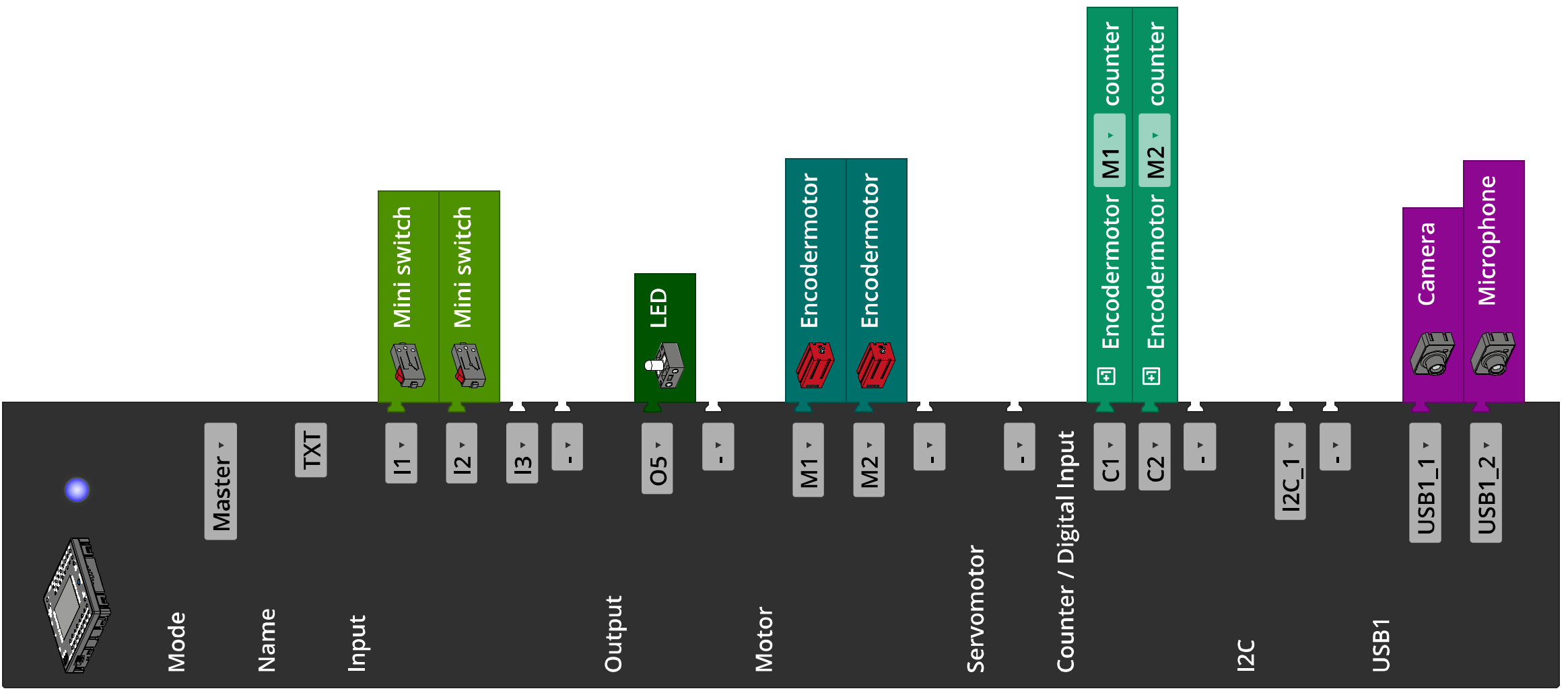
# Alarmanlage

## Konstruktionsaufgabe

Siehe Bauanleitung.

## Programmieraufgaben

Konfiguration der Sensoren und Aktoren:



Für die Lösung von Experimentieraufgabe 2 wird die „Voice Control“-App (für iOS oder Android) benötigt. Die App muss für die Spracherkennung mit dem Internet verbunden sein und (über Bluetooth oder WLAN) mit dem Controller verbunden werden.

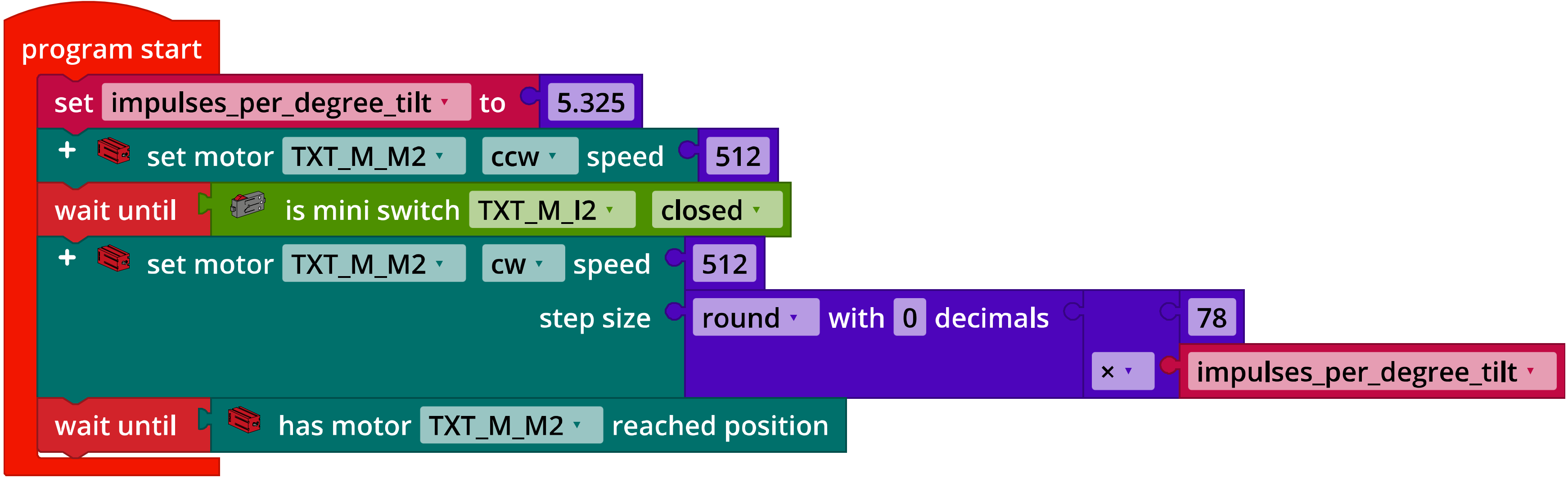
**1. Grundstellung**

1a. Jede Umdrehung der Motorachse entspricht einer Umdrehung der Schnecke, und jede Umdrehung der Schnecke dreht das Z30 um einen Zahn weiter. Für eine 90°-Drehung des Z30 werden also 30/4 Umdrehungen der Motorachse benötigt. Das entspricht (bei 63,9 Encoder-Impulsen je Achsumdrehung)

Encoder-Impulsen (oder 5,325 Impulsen je Winkelgrad).

Tatsächlich wird der Endlagentaster nicht genau nach einer Vierteldrehung ausgelöst; die Kamera muss lediglich etwa 78° gedreht werden. Das entspricht Encoder-Impulsen.

Programm (Beispiel):



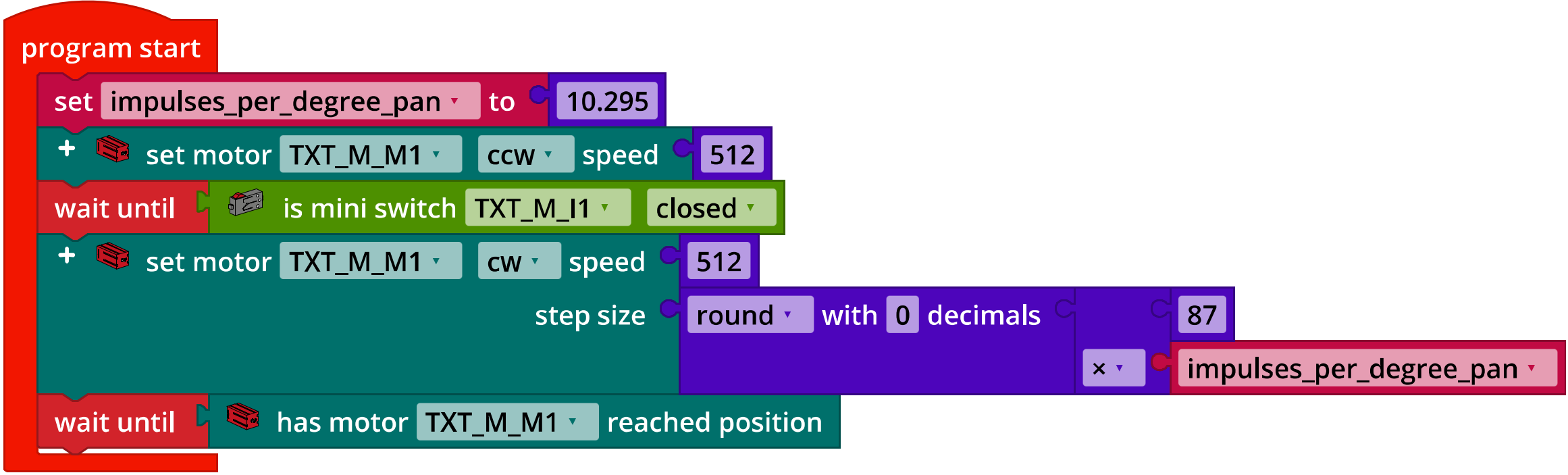
*IoT\_Init\_Camera\_Tilt.ft*

1b. Der Drehkranz hat 58 Zähne; eine 90°-Drehung benötigt also 58/4 Umdrehungen der Motorachse. Das entspricht

Encoder-Impulsen (oder 10,295 Impulsen je Winkelgrad).

Auch hier wird der Endlagentaster nicht erst nach einer 90°-Drehung ausgelöst, bereits bei ca. 3° ausgelöst; die Kamera muss lediglich um etwa 87° gedreht werden. Das entspricht Encoder-Impulsen.

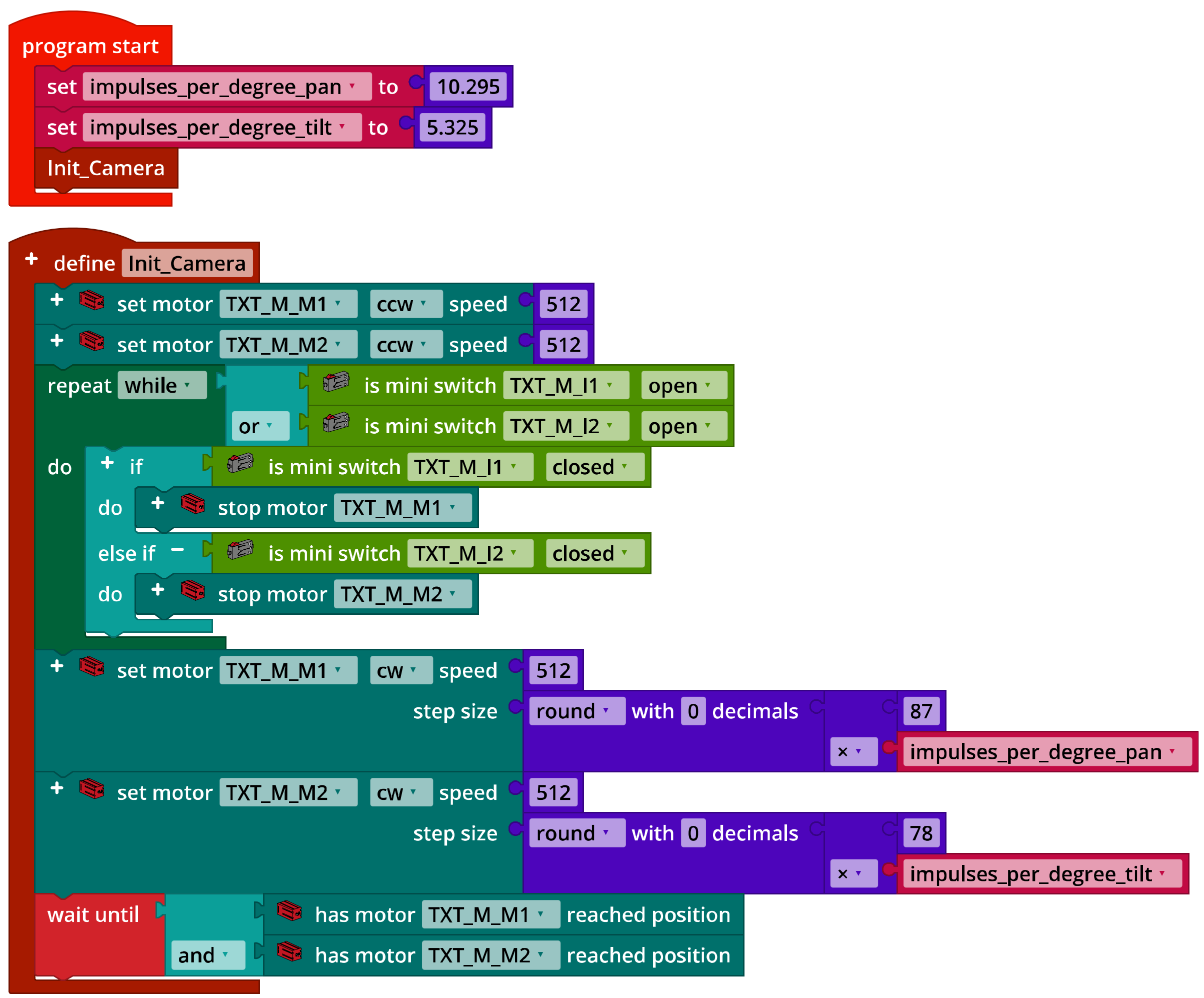
Programm (Beispiel):



*IoT\_Init\_Camera\_Pan.ft*

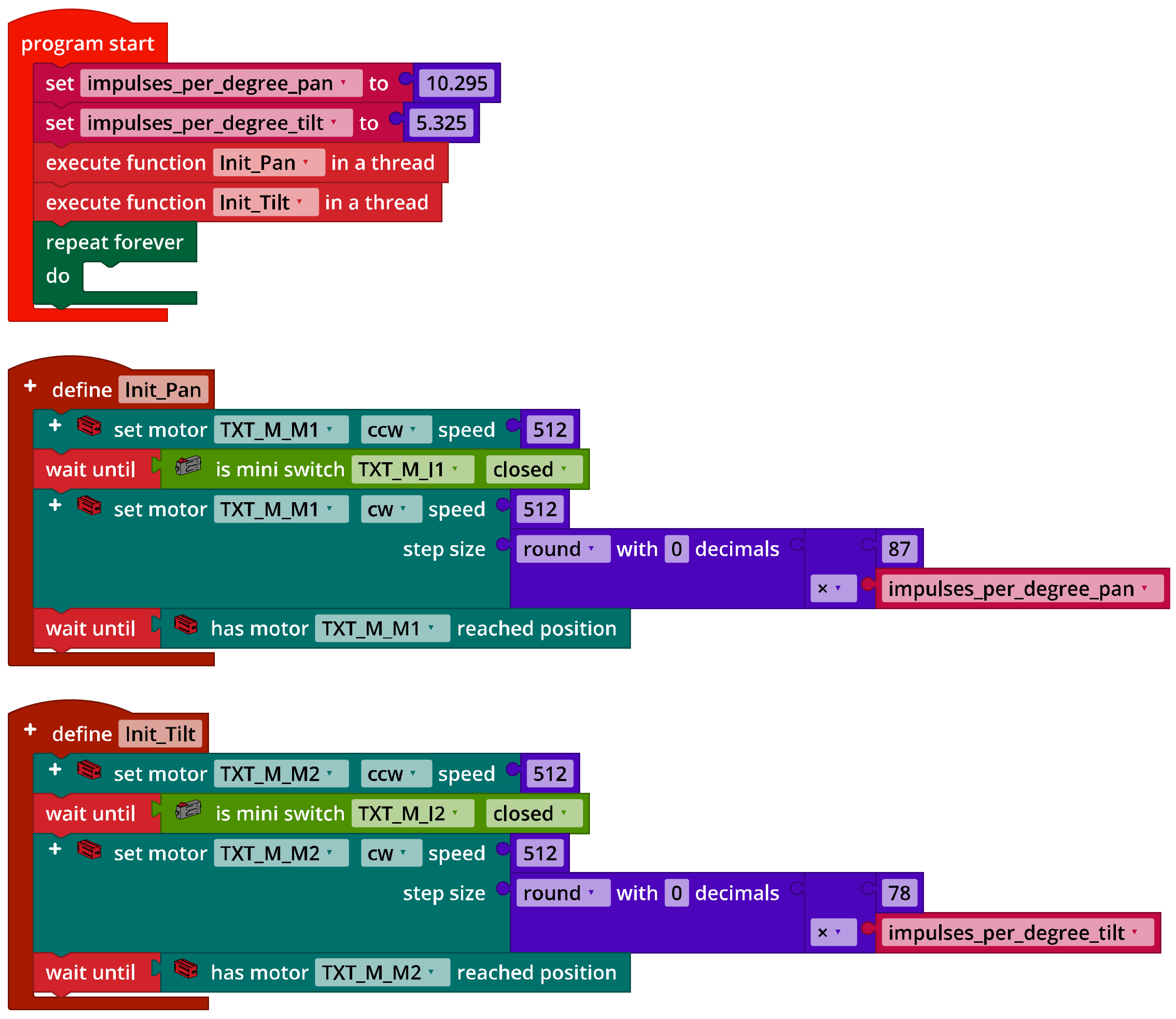
1c. Bei der Zusammenführung der beiden Programme können die beiden Bewe­gungen parallelisiert und die Initialisierungszeit damit erheblich verkürzt werden. Dafür gibt es zwei Lösungswege: Ausführung der beiden Funktionen als parallele Threads oder Integration in einer Funktion.

Programm Lösungsweg A – eine Funktion (Beispiel):



*IoT\_Init\_Camera\_A.ft*

Programm Lösungsweg B – parallele Threads (Beispiel):



*IoT\_Init\_Camera\_B.ft*

**2. Kameraüberwachung**

Programm (Beispiel):

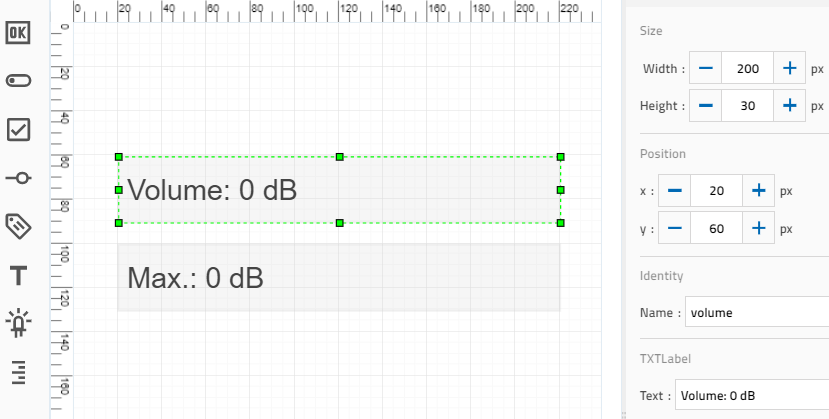
Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

*IoT\_Surveillance\_Camera.ft*

**3. Geräuschaktivierung**

3a. Konfiguration des TXT-Display (Beispiel):

Display-Konfiguration

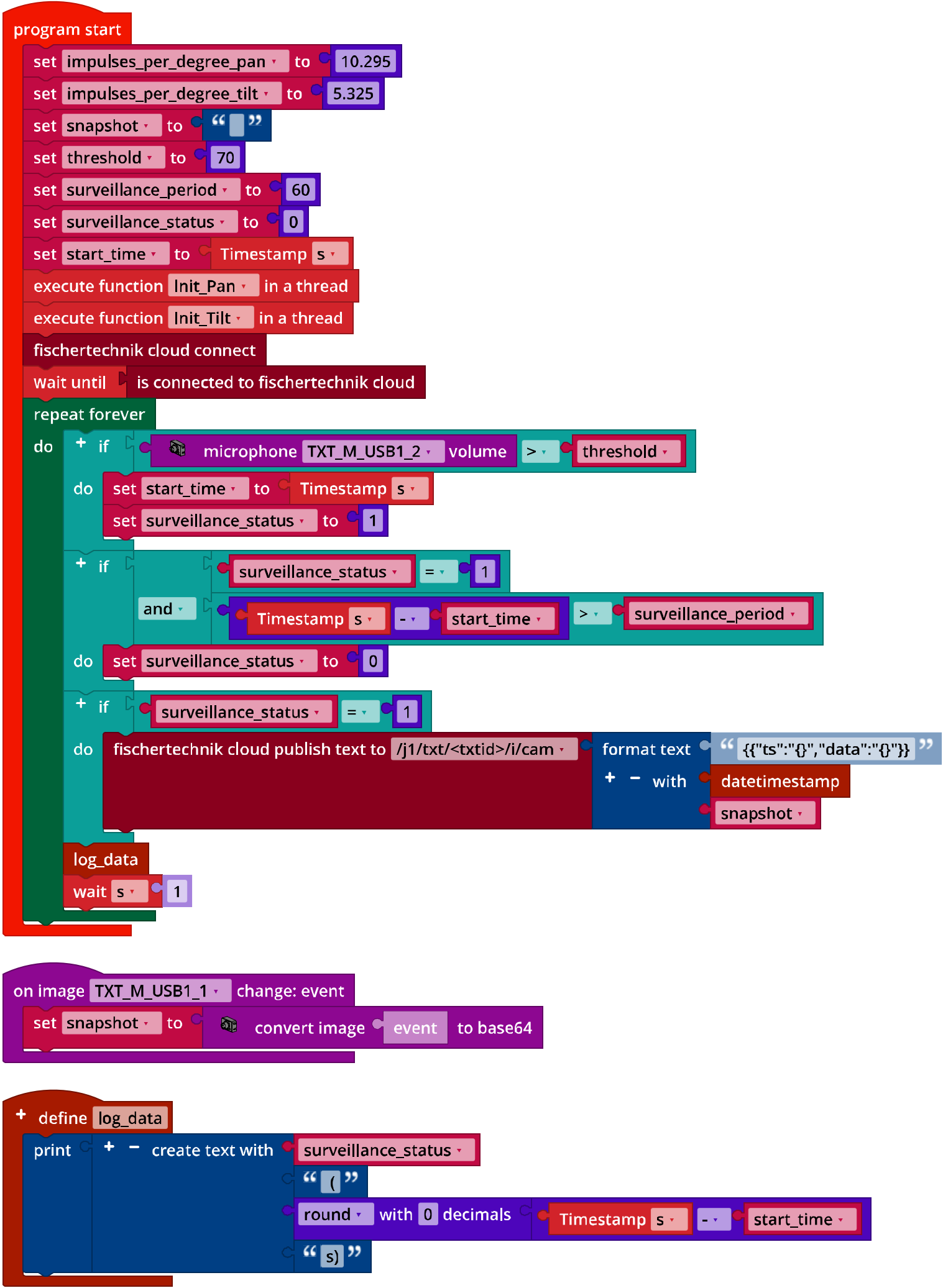
Programm (Beispiel):

Ein Bild, das Text, Screenshot, Monitor, Bildschirm enthält.

Automatisch generierte Beschreibung

*IoT\_Microphone\_Level.ft*

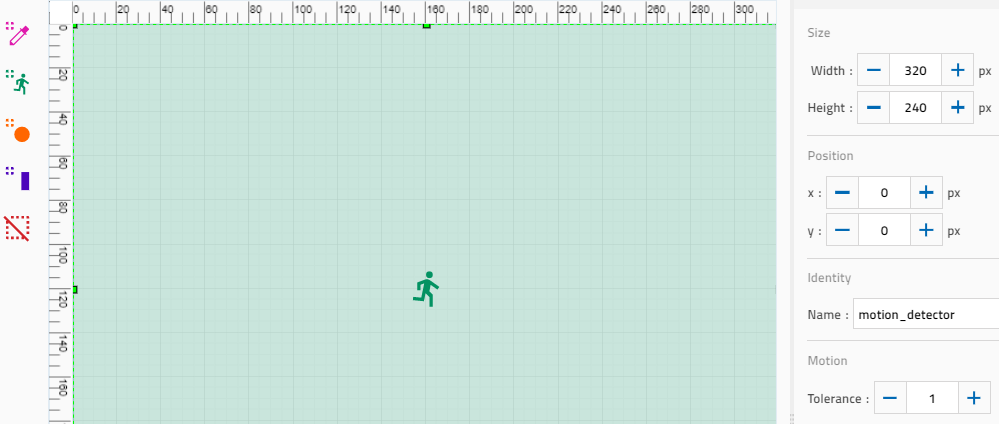
3b. Programm mit Lautstärke-Schwelle 70 dB (Beispiel):



*IoT\_Surveillance\_Camera\_Noise\_Detection.ft*

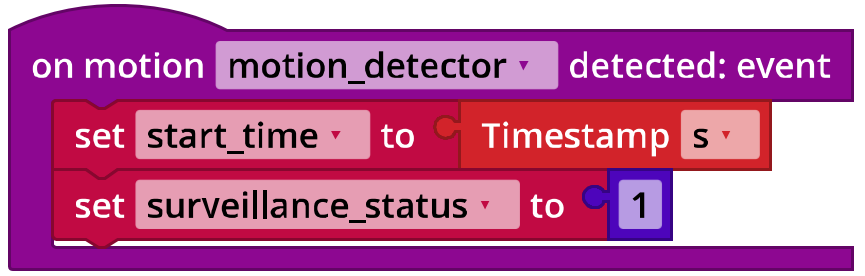
**4. Bewegungserkennung**

Die Bewegungserkennung wird über das gesamte Kamerafenster definiert. Über die „Toleranz“ kann im Inspektor die Empfindlichkeit der Erkennung auf einen Wert von 0 bis 1 eingestellt werden:



Konfiguration Bewegungserkennung

4a. Programmauszug (Beispiel):



*IoT\_Surveillance\_Camera\_Motion\_Detection.ft*

Das Erkennen einer Bewegung wird in einem Flag („surveillance\_status“) gespeichert, ebenso der Zeitpunkt der Erkennung.

4b. Die Funktion „blink“ wird zu Beginn des Hauptprogramms als Thread gestartet:

Ein Bild, das Text, Screenshot, Schild enthält.

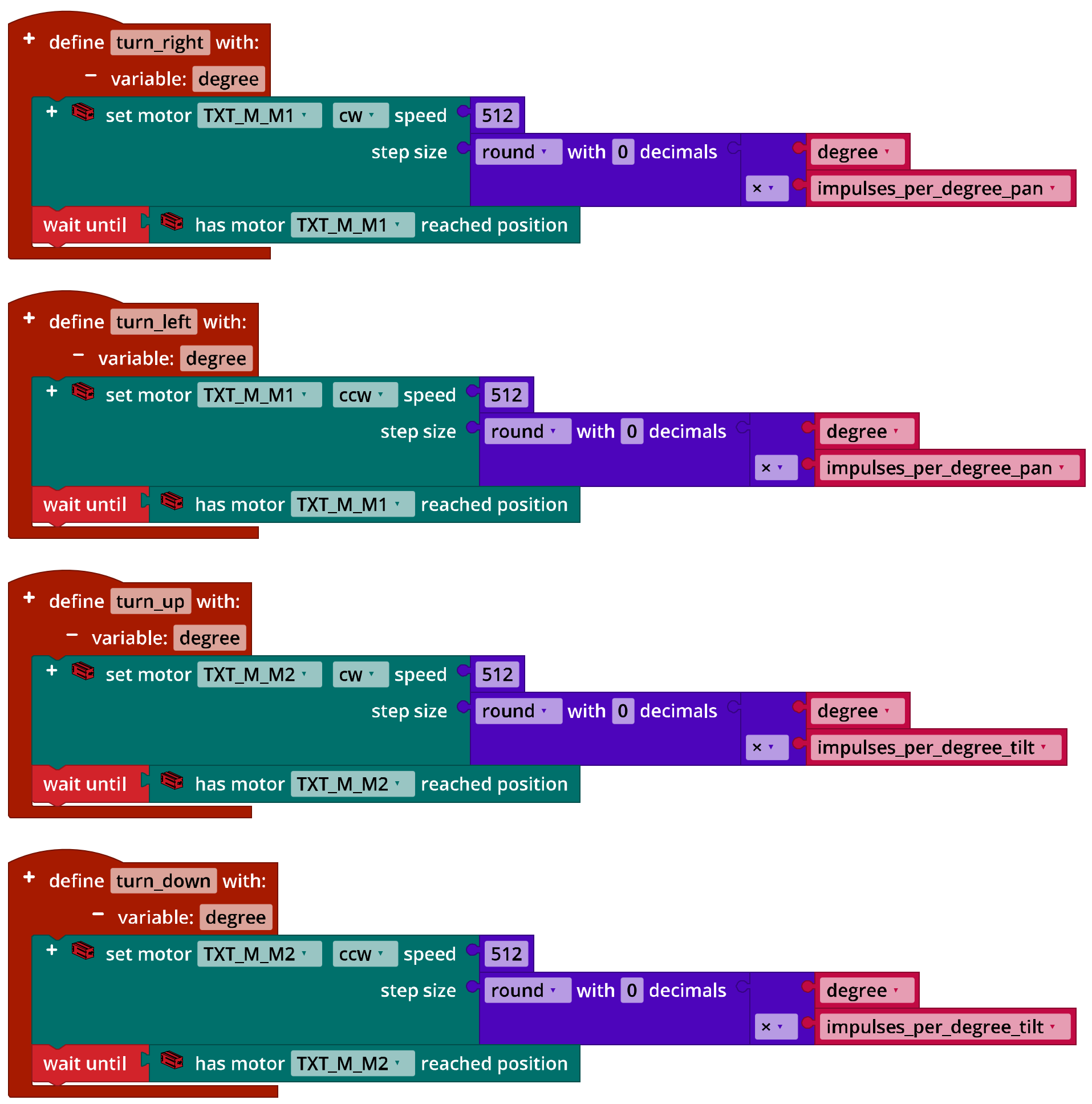
Automatisch generierte Beschreibung

*IoT\_Surveillance\_Camera\_Motion\_Detection.ft*

## Experimentieraufgaben

**1. Sprachsteuerung**

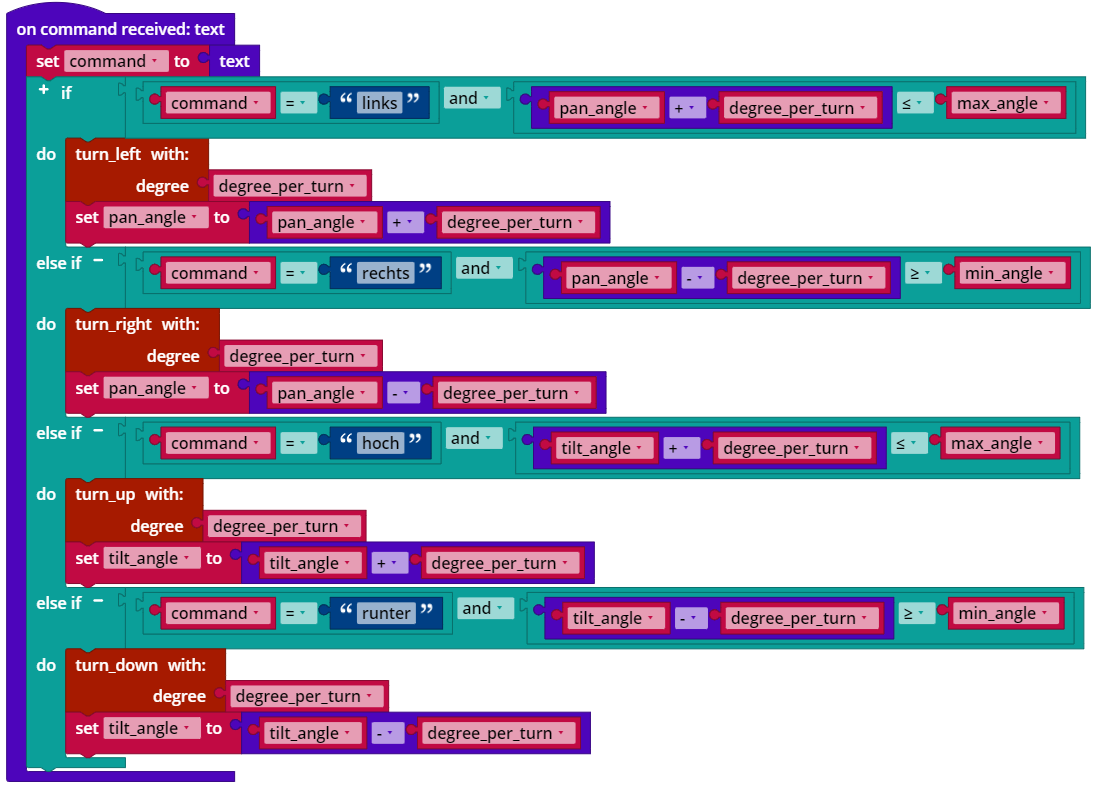
1a. Steuerungsfunktionen (Beispiel):



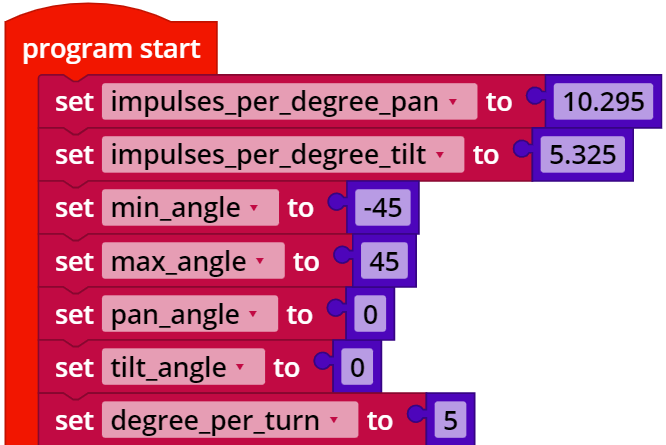
*IoT\_Camera\_Motion\_Control.ft*

Ein zu weites Drehen der Kamera lässt sich am einfachsten verhindern, indem der Drehwinkel in horizontaler und vertikaler Richtung auf jeweils insgesamt ±90° begrenzt wird. Setzt man beide Auslenkungswinkel nach der Initialisierung der Grundstellung auf 0°, so ist das im Programm leicht zu überprüfen.

1b. Programmauszug: Steuerung der Kamera (Beispiel):

*IoT\_Surveillance\_Camera\_Voice\_Control.ft*

Im Hauptprogramm werden zu Beginn die folgenden Variablen initialisiert:



*IoT\_Surveillance\_Camera\_Voice\_Control.ft*

Der maximale und der minimale Auslenkungswinkel sind ggf. an die Konstruktion anzu­passen (Reichweite der Verkabelung).

**2. Steuerung über das Cloud-Dashboard**

Die Steuerungsknöpfe des Dashboards liefern die Kommandos „stop“, „home“, „relmove\_left“, „relmove\_right“, „relmove\_up“ und „relmove\_down“. Die „relmove“-Kommandos entsprechen den vier Steuerfunktionen aus Experi­mentieraufgabe 1. Wird vom IoT-Server ein Wert „degree“ (2, 5, 10 oder 20°) übermittelt, muss die Dreh­winkelvorgabe im Programm angepasst werden. Das Kommando „home“ bewegt die Kamera in die Grundstellung; für das Kommando „stop“ ist eine Funktion zu ergänzen, die die beiden Motoren stoppt.

Die Abfrage der Steuerkommandos kann aus der Lösung zu Experimentieraufgabe 1 übernommen werden.

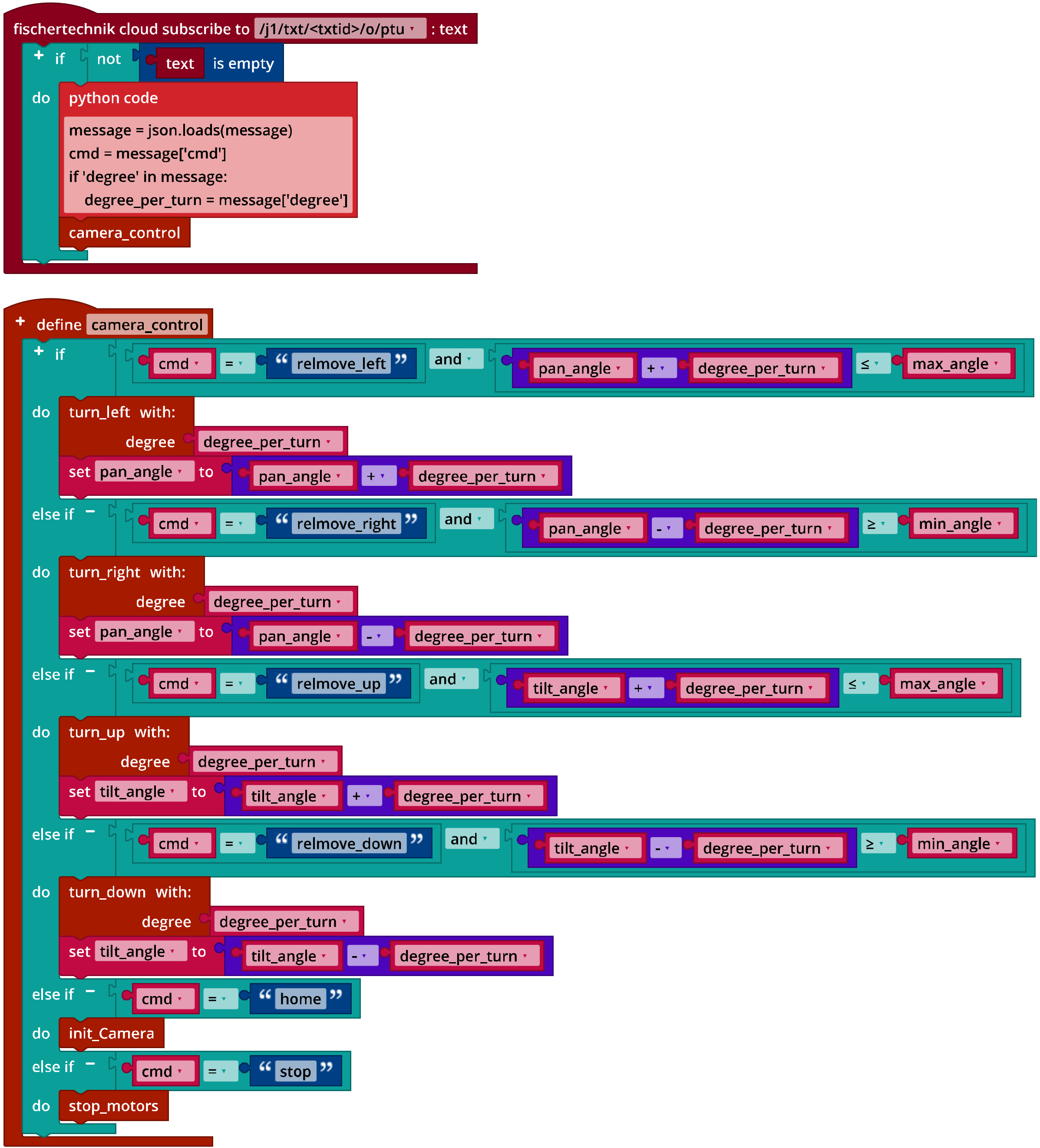
Funktion „stop\_motors“ (Beispiel):

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

*IoT\_Surveillance\_Camera\_Dashboard\_Control.ft*

Programmauszug: Steuerung der Kamera (Beispiel):



*IoT\_Surveillance\_Camera\_Dashboard\_Control.ft*

Anlagen

# Aufgabe 3: Alarmanlage

## Erforderliches Material

* PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
* USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
* fischertechnik-App „Voice Control“
* Hilfsprogramm „IoT\_Test\_Dashboard\_Control.ft”
* Account in der fischertechnik-Cloud ([www.fischertechnik-cloud.com](http://www.fischertechnik-cloud.com))