Tous les datagrammes sont représentés par des strings dans lesquelles les valeurs sont séparées par des virgules

Les relations Application Android -> Boitier de commande :

L’application envoie un datagramme composé d’une liste d’informations

ID\_COMMANDE : L’identification de la commande envoyée

MODE : Un identifiant de type de commande

ACCELERATION : L’acceleration du moteur du plateau souhaitée

VITESSE : La vitesse du moteur du plateau

NOMBRE\_DE\_PAS\_TABLE : Le nombre de pas de la table

DIRECTION : Le sens de rotation du moteur du plateau (0-1)

CHOIX\_ROTATION : Le type de rotation souhaitée

NOMBRE\_DE\_TOUR ou TEMPS\_DE\_ROTATION : Le temps de rotation du moteur ou le nombre de tour (dépendant du CHOIX\_ROTATION)

NOMBRE\_DE\_PHOTOS : Le nombre de photos souhaitées pour la prise de vue

NOMBRE\_DE\_CAMERAS : Le nombre de caméras à déclancher lors de chaque arrêt

PAUSE\_ENTRE\_CAMERAS : Le temps de pause entre les photos

FOCUS\_STACKING : Le nombre de prises de vues avec niveau de focus différents par appareil photo et par arrêt.

Lors de la réception du datagramme, le bloc de commande va récupérer les informations importantes et créer une liste d’instructions élémentaires à faire et à envoyer au(x) périphérique(s).

Par exemple, si le mode est à 1 (mode temps réel), la liste consistera en 1 seule instruction élémentaire qui sera envoyée au moteur du plateau pour qu’il tourne.

Le bloc de commande renvoie 3 types d’informations au téléphone :

* 1 datagramme lors de la création d’une instruction élémentaire :

La chaine de caractère ‘creation’

ID\_COMMANDE : l’identifiant de la commande concernée

ID\_INSTRUCTION : l’identifiant de l’instruction dans la liste d’instruction de cette commande

Le nom du périphérique concerné (ex : ‘moteur’)

Le datagramme qui sera envoyé à ce périphérique

* 1 datagramme lors du démarrage d’une instruction élémentaire :

La chaine de caractère ‘en cours

ID\_COMMANDE : l’identifiant de la commande concernée

ID\_INSTRUCTION : l’identifiant de l’instruction dans la liste d’instruction de cette commande

* 1 datagramme lors de la fin d’une commande

La chaine de caractère ‘fini’

ID\_COMMANDE : l’identifiant de la commande concernée

Les relations bloc de commande – périphériques :

Les échanges entre le bloc de commande et un périphérique s’effectue par le biais d’une chaine de caractère commune, sur laquelle chaque coté peut écrire et lire.

Un périphérique doit écrire son statut de fonctionnement dans cette chaine :

Un périphérique qui écrit ‘0’ dans sa chaine est un périphérique disponible. Un périphérique doit donc écrire ‘0’ lors de son initialisation.

Lorsque le bloc de commande devra envoyer une instruction élémentaire à un périphérique, il écrira un datagramme comportant toutes les informations nécessaires au bon fonctionnement du périphérique.

Une fois que le périphérique aura pris en compte l’instruction, il se mettra en fonctionnement et écrit ‘1’ dans la chaine, signifiant qu’il est au travail. Le bloc de commande ne renverra plus d’instruction tant qu’un périphérique est en fonctionnement.

Une fois la tache terminée, le périphérique écrit ‘2’ dans sa chaine, indiquant qu’il a terminé l’instruction. Le bloc de commande, une fois l’information reçu, se charge d’écrire ‘0’ dans le périphérique pour le libérer. Un périphérique ne doit jamais écrire lui-même ‘0’, sauf dans sin initialisation.

Facultatif :

Le bloc de commande écrira ‘3’ dans un périphérique pour déclencher une pause, ou pour l’arrêter. Un périphérique n’est pas obligé de traiter la pause, mais le bloc de commande écrira quand même ‘3’.

* Datagramme moteur :

ACCELERATION

VITESSE

DIRECTION

CHOIX\_ROTATION

NOMBRE\_DE\_TOUR ou TEMPS\_DE\_ROTATION

* Datagramme caméra focus stacking

TEMPS\_PAUSE : le temps de pause à effectuer après avoir pris la photo

NOMBRE\_DE\_FOCUS : le nombre de photo à prendre avec des focus différents

PAS\_ENTRE\_PHOTOS : une liste comprenant le nombre de pas à effectuer entre chaque changement de focus. La longueur de la liste doit être la même que le NOMBRE\_DE\_FOCUS