YP-Spur 走行制御コマンド

単位はそれぞれ，位置 m，角度 rad，時間 s，速度 m/s ，角速度 rad/s を用いる

初期化

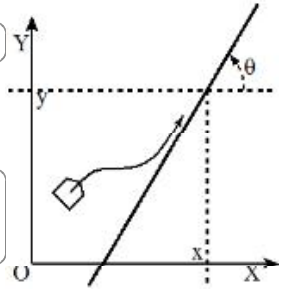
```
void Spur_init( void )
```

他のコマンドを使用する前に実行する。

直線追従

```
void Spur_line_GL( double x, double y, double th )
void Spur_line_LC( double x, double y, double th )
void Spur_line_FS( double x, double y, double th )
```

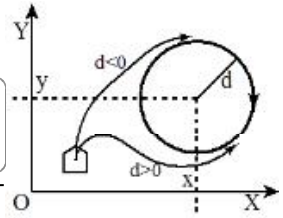
それぞれの座標系で(x,y)を通り角度 th の直線に追従する。



円弧追従

```
void Spur_circle_GL( double x, double y, double d)
void Spur_circle_LC( double x, double y, double d)
void Spur_circle_FS( double x, double y, double d)
```

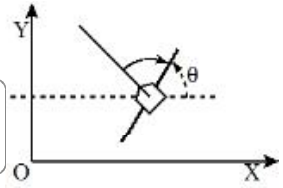
それぞれの座標系で(x,y)を中心とする半径 d の円弧に追従する。d ＜ 0 で時計回り，d ＞ 0 で反時計回りを表す。



回転

```
void Spur_spin_GL( double th )
void Spur_spin_LC( double th )
void Spur_spin_FS( double th )
```

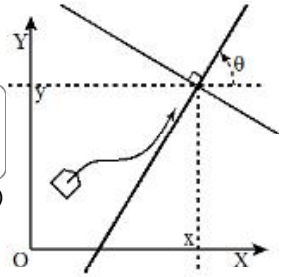
それぞれの座標系で姿勢を角度 th に制御する。



直線上での停止

```
void Spur_stop_line_GL( double x, double y, double th )
void Spur_stop_line_LC( double x, double y, double th )
void Spur_stop_line_FS( double x, double y, double th )
```

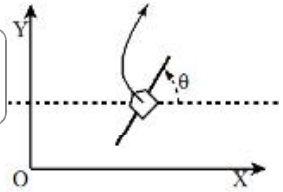
それぞれの座標系で(x,y)を通り角度 th の直線に追従し，(x,y)を通り角度 th に垂直な直線上で，角度 th を向いて停止する。



方位指令の走行

```
void Spur_orient_GL( double th )
void Spur_orient_LC( double th )
void Spur_orient_FS( double th )
```

角度 th の方向に走行する。



停止･フリー

```
void Spur_stop( void )
void Spur_free( void )
```

stop は最大加速度で減速し停止する。
free は速度制御を停止し，摩擦補償のみ行う。

状態取得コマンド

現在位置取得

```
void Spur_get_pos_GL( double *x, double *y, double *th )
void Spur_get_pos_LC( double *x, double *y, double *th )
```

それぞれの座標系でのロボットの位置を取得する。

現在速度取得

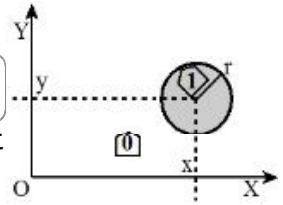
```
void Spur_get_vel( double *v, double *w )
```

ロボットの現在の速度・角速度を取得する。

位置判定

```
int Spur_near_pos_GL( double x, double y, double r)
int Spur_near_pos_LC( double x, double y, double r)
```

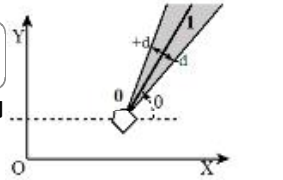
それぞれの座標系で(x,y)を中心とする半径 r の円より内側にロボットがいれば 1 を返す。



角度判定

```
int Spur_near_ang_GL( double th, double d )
int Spur_near_ang_LC( double th, double d )
```

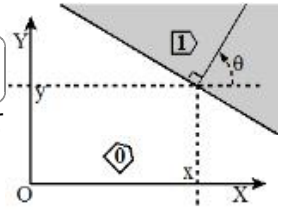
それぞれの座標系でロボットが角度 th から± d の範囲より内側を向いているなら 1 を返す。



領域判定

```
int Spur_over_line_GL( double x, double y, double th )
int Spur_over_line_LC( double x, double y, double th )
```

それぞれの座標系で(x,y)を通り角度 th に垂直な直線を越えていれば 1 を返す。



パラメータ操作コマンド

座標系設定

```
void Spur_set_pos_GL( double x, double y, double th )
void Spur_set_pos_LC( double x, double y, double th )
```

ロボットの位置・姿勢が(x,y,th)になるように座標系を変更する。

座標系修正

```
void Spur_adjust_pos_GL( double x, double y, double th )
void Spur_adjust_pos_LC( double x, double y, double th )
void Spur_adjust_pos_FS( double x, double y, double th )
```

ロボットの位置・姿勢が(x,y,th)だったことが判ったとき、そうなるように座標系を修正する。これにあわせて、実行中の走行制御コマンドも正しい位置・姿勢に修正される。

最大速度指定

```
void Spur_set_vel( double v )
void Spur_set_angvel( double w )
```

ロボットの速度・角速度を指定する。

最大加速度指定

```
void Spur_set_accel( double v )
void Spur_set_angaccel( double w )
```

ロボットの加速度・角加速度を指定する。

地面の傾き指定

```
void Spur_tile_GL( double d, double t )
void Spur_tile_LC( double d, double t )
void Spur_tile_FS( double d, double t )
```

坂の勾配方向 d と傾斜角 t を指定する。

その他

速度直接入力

```
void Spur_vel( double v, double w )
```

ロボットの並進速度・角速度を直接指定する。

タイヤ軸角度サーボ

```
void YP_set_wheel_vel( double wr, double wl )
void YP_set_wheel_accel( double or, double ol )
void YP_wheel_ang( double ar, double al )
void YP_wheel_vel( double r, double l )
```

set_wheel_vel, accel でタイヤ軸の角速度・各加速度を指定し、wheel_ang, vel で角度指令・速度指令値を与える。

トルク制御

```
void Spur_get_force( double *f, double *t )
void YP_get_wheel_torque( double *tr, double *tl )
void YP_wheel_torque( double tr, double tl )
```

get_force で、ロボットが外界に与える並進の力・回転のトルクの推定値を取得。
get_wheel_torque で各タイヤの出力トルクの推定値を取得し、wheel_torque で出力トルクの指令値を与える。

緊急停止

```
void Spur_freeze( void )
void Spur_unfreeze( void )
int Spur_isfreeze( void )
```

freeze で緊急停止，unfreeze で解除する。
isfreeze は緊急停止中なら 1 を返す。
緊急停止中は，他のコマンドを受け付けるが，走行制御は行わない。解除すると，最新のコマンドの走行制御が再開する。

備考：
返値が負の場合はメッセージ通信の失敗を意味する。再び Spur_init 関数を呼び出すことで再度通信を開始しようとする。メッセージ通信が失敗した状態になると、YPSpur_get_error_state()関数が真を返すようになる。
本稿は，YPSpur-1.13.3(2012-06-05)の版に準拠している。

AWD