Velocity mode

We have two kinds of velocity mode. The first is Joint velocity mode and the other is End-effector velocity mode.

- ⚠ Please use velocity mode function carefully.
- ⚠ If the command does not appropriately set, robot could be rush.

1. Joint velocity mode

In the Joint velocity mode, you can send the joints' velocity command and the joints will reach to the speed.

1-1 ContinueVJog()

This function can start the joint velocity mode.

Syntax

```
bool ContinueVJog ()
```

Parameter

`void` No input values required

Return

`bool:` True: Command accepted; False: Command rejected

1-2 SetContinueVJog ()

This function can send the velocity command.

Syntax1

```
bool SetContinueVJog(float v1, float v2, float v3, float v4, float v5, float v6)
```

Parameter

`V1:` The first motor velocity command and the unit is deg/s.

`V2:` The second motor velocity command and the unit is deg/s.

`V3:` The third motor velocity command and the unit is deg/s.

`V4:` The fourth motor velocity command and the unit is deg/s.

`V5:` The fifth motor velocity command and the unit is deg/s.

`V6:` The sixth motor velocity command and the unit is deg/s.

Return

`bool:` True: Command accepted; False: Command rejected.

Syntax2

```
bool SetContinueVJog(float[])
```

Parameter

`float[]:` A 6x1 array. They are 6 motors velocity command, the unit is deg/s. The

first parameter means first motor velocity command and so on.

Return

`bool:` True: Command accepted; False: Command rejected.

1-3 StopContinueVmode()

This function stops the joint velocity mode.

Syntax

```
bool StopContinueVmode()
```

Parameter

`void` No input values required

Return

`bool:` True: Command accepted; False: Command rejected.

2. End-Effector velocity mode

In the End-Effector velocity mode, you can send the end-effector velocity command and the end-effector will reach to the speed.

2-1 ContinueVLine()

This function can start the End-Effector velocity mode.

Syntax

```
bool ContinueVLine(int stopTimeMs, int breakLoopMs)
```

Parameter

`stopTimeMs:` If this function finds you do not renew the velocity data after this value time in millisecond(ms), it will set the speed down to zero.

`breakLoopMs:` If this function finds you do not renew the velocity data after this value time in millisecond(ms), it will stop the End-Effector velocity mode.

Return

`bool:` True: Command accepted; False: Command rejected.

Note

The value stopTimeMs should smaller than the value breakLoopMs.

2-2 SetContinueVLine ()

This function can send the velocity command.

Syntax1

```
bool SetContinueVLine(float vx, float vy, float vz, float wx, float wy, float wz)
bool SetContinueVLine(float[])
```

Parameter

`vx`: the x-direction velocity and the unit is mm/ms.

`vy`: the y-direction velocity and the unit is mm/ms.

`vz`: the z-direction velocity and the unit is mm/ms.

`wx`: the x-direction rotation and the unit is deg/ms.

`wy`: the y-direction rotation and the unit is deg/ms.

`wz`: the z-direction rotation and the unit is deg/ms.

Return

`bool` True: Command accepted; False: Command rejected.

Syntax2

```
bool SetContinueVLine(float[])
```

Parameter

`float[]:` A 6x1 array. They are 6 direction velocity commands on the Syntax1.

Return

`bool` True: Command accepted; False: Command rejected.

2-3 SuspendContinueVmode ()

This function sets all speed down to zero and it waits for next command coming.

Syntax

```
bool SuspendContinueVmode ()
```

Parameter

`void` No input values required

Return

`bool:` True: Command accepted; False: Command rejected.

2-4 StopContinueVmode ()

This function stops the End-Effector velocity mode.

Syntax

```
bool StopContinueVmode ()
```

Parameter

`void` No input values required

Return

`bool:` True: Command accepted; False: Command rejected.