DOCUMENTATION ISG-kernel

Manual
Cycles - Additional Cycles

Short Description:
CYCLES-ADC

ISG Industrielle Steuerungstechnik GmbH
STEP, Gropiusplatz 10
D-70563 Stuttgart

www.isg-stuttgart.de
support@isg-stuttgart.de

Documentation version: 1.0
26/02/2024

Preface

Legal information

This documentation was produced with utmost care. The products and scope of functions described are under continuous development. We reserve the right to revise and amend the documentation at any time and without prior notice.

No claims may be made for products which have already been delivered if such claims are based on the specifications, figures and descriptions contained in this documentation.

Personnel qualifications

This description is solely intended for skilled technicians who were trained in control, automation and drive systems and who are familiar with the applicable standards, the relevant documentation and the machining application.

It is absolutely vital to refer to this documentation, the instructions below and the explanations to carry out installation and commissioning work. Skilled technicians are under the obligation to use the documentation duly published for every installation and commissioning operation.

Skilled technicians must ensure that the application or use of the products described fulfil all safety requirements including all applicable laws, regulations, provisions and standards.

Further information

Links below (DE)

https://www.isg-stuttgart.de/produkte/softwareprodukte/isg-kernel/dokumente-und-downloads

or (EN)

https://www.isg-stuttgart.de/en/products/softwareproducts/isg-kernel/documents-and-downloads

contains further information on messages generated in the NC kernel, online help, PLC libraries, tools, etc. in addition to the current documentation.

Disclaimer

It is forbidden to make any changes to the software configuration which are not contained in the options described in this documentation.

Trade marks and patents

The name ISG®, ISG kernel®, ISG virtuos®, ISG dirigent® and the associated logos are registered and licensed trade marks of ISG Industrielle Steuerungstechnik GmbH.

The use of other trade marks or logos contained in this documentation by third parties may result in a violation of the rights of the respective trade mark owners.

Copyright

General and safety instructions

Icons used and their meanings

This documentation uses the following icons next to the safety instruction and the associated text. Please read the (safety) instructions carefully and comply with them at all times.

Icons in explanatory text

➢ Indicates an action.

⇨ Indicates an action statement.

⚠ DANGER

Acute danger to life!

If you fail to comply with the safety instruction next to this icon, there is immediate danger to human life and health.

⚠ CAUTION

Personal injury and damage to machines!

If you fail to comply with the safety instruction next to this icon, it may result in personal injury or damage to machines.

Attention

Restriction or error

This icon describes restrictions or warns of errors.

Notice

Tips and other notes

This icon indicates information to assist in general understanding or to provide additional information.

Example

General example

Example that clarifies the text.

Programing Example

NC programming example

Programming example (complete NC program or program sequence) of the described function or NC command.

Release Note

Specific version information

Optional or restricted function. The availability of this function depends on the configuration and the scope of the version.

Contents

List of figures

1 Cycles - Introduction

Notice

Cycles are additional options and subject to the purchase of a license.

General information

Cycle call

ISG cycles are called using a cycle call:

```
L CYCLE[ NAME="..." @P1 = .. @P2 = .. ...]
```

A cycle is called by specifying the cycle name. It is also possible to parameterise the cycle input parameters to modify cycle behaviour to a special application.

This documentation contains a separate subsection for each cycle where cycle behaviour is described in greater detail. It also contains a list of the input parameters used for the cycle. Finally, a simple programming example describes how to call the cycle.

Selecting the cycle plane

A cycle is programmed independently of the currently valid plane (G17, G18, G19) and independently of the axis names configured in the NC channel.

In the cycle documentation, the axes are described by the following names for the sake of better legibility:

- The X axis describes the 1st main axis
- The Y axis describes the 2nd main axis
- The Z axis describes the 3rd main axis
- A axis is the designation for the rotary axis about the 1st main axis
- B axis is the designation for the rotary axis about the 2nd main axis
- C axis is the designation for the rotary axis about the 3rd main axis

Notice

Cycles can also be used in offset and rotated coordinate systems. These coordinate systems should only be defined using the #CS command. The #ROTATION command is not suitable for use in combination with cycles.

1.1 Note on installing the cycles

Content of the cycle packet

The cycle packet contains the following files and directories:

- Set-up file to execute the cycle set-up.
- Dependency graph: Describes the dependencies (subroutine calls) for each of the cycles.
- Release notes: Description of relevant changes which the release contains.
- Documentation index: Contains the cycle documentation.
- Index of error messages: Contains the cycle error messages.
- Condensed index of cycles: This contains the CNC cycle and example call programs. When the set-up is executed, the files are unpacked and saved on the specified path.

Executing the set-up

To install the cycles, execute the file *isg-cnc-cycles-setup.exe* contained in the cycle packet.

When the cycle set-up is executed, the CNC cycles and the example call programs are unpacked and saved on the specified path.

To execute the cycles, the specified path must be saved in the controller as the subroutine path.

Attention

When the cycle set-up is executed the contents of the previously installed cycle packet are overwritten.

Integrating error messages

The error files must be saved in the TwinCAT set-up to ensure that the correct error messages are output if CNC cycles are incorrectly executed.

The file *TcCncCycleErrors.xml* in the error directory (*error_files*) in the TwinCAT set-up directory (here *C:\TwinCat*) is saved on the following path in order to output error message in the HMI: *C:\TwinCAT\3.1\Target\Resource*.

The file *err_text_cycles_eng.txt* (or *err_text_cycles.txt*) in the error directory (*error_files*) in the TwinCAT set-up directory (here *C:\TwinCat*) is saved on the following path in order to output error messages correctly in the log file: *C:\TwinCAT\3.1\Components\Mc\CNC\Diagnostics*.

Alternatively, the name of the error message file can also be modified by the channel parameter P-STUP-00200.

2 Overview

Notice

Cycles are additional options and subject to the purchase of a license.

Additional cycles can be used provided one or the three cycle main licenses (Machining, Measurement and Calibration and calibrating, Machine Calibration) is installed.

Task

In addition to existing cycles, the additional cycles offer the following scope of functions:

1. The positioning cycles [▸ 10] are intended to calculate and pre-position the tool. Normally, the cycles are called in conjunction with machining, measuring or calibration cycles.
2. The calculation cycles [▸ 21] are used to execute complex calculations that require the user to have a lot of technical knowledge and carry out extensive programming work.
3. In general, the machine response can be optimised by means of channel parameters using the High Speed Settings cycle [▸ 36]. This allows you to influence the surface quality, speed and accuracy for a specific machine.

3 Positioning cycles

3.1 Introduction

Notice

Cycles are additional options and subject to the purchase of a license.

Overview

The positioning cycles are intended to calculate and pre-position the tool. Normally, the cycles are called in conjunction with machining, measuring or calibration cycles.

Possible applications

The following applications are conceivable:

- Calculate a plane by specifying reference points
- Rotate the machining coordinate system to an inclined plane
- Rotate the tool to an inclined plane
- Using predefined call patterns to call a cycle

Programming

The cycles are called with the L CYCLE[…] function and the required parameters are taken directly from the NC program. A more detailed description of the call is provided in the subsections for each of the cycles.

3.2 SysPosRotationToPlane - Rotate to plane

This cycle rotates the tool and/or the machining coordinate system [MCS] to a plane defined by the input parameters.

The required plane is defined by rotation angle or reference points. Optionally, the resulting machining coordinate system can be checked in advance by an output. This cycle can be used after a teach-in cycle to rotate to the measured MCS.

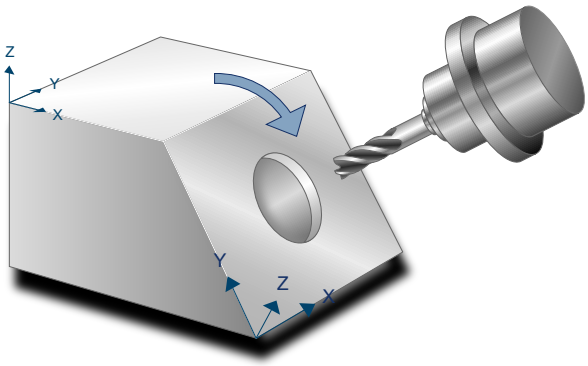

Fig. 1: SysPosRotationToPlane

3.2.1 Process

Notice

If the tool needs to be rotated to the defined plane, a complete kinematic transformation must be active before the cycle.

At the start of the cycle, the tool is retracted along the Z axis by the section set in @P2.

The following processes can be set by the rotation mode @P1_

- @P1=1: The machining coordinate system and tool are rotated to the defined plane
- @P1=2: Only the tool is rotated to the defined plane
- @P1=3: Only the machining coordinate system is rotated to the defined plane
- @P1=4: No movement occurs / change in MCS This mode can be used to calculate and output the MCS resulting from the input parameters.

Once the machining coordinate system is rotated (@P1=1 or @P1=3), after the cycle a rotated MCS with zero offset is active on the origin defined by @P3@P5. This can be deselected by #CS OFF.

To check the correct definition of the required plane, the resulting MCS can be output to a text file by @P20=1. The calculated axes of the MCS represented in the MCS are then written to the text file SysPosRotationToPlaneResult.txt.

3.2.2 Definition of the plane by reference points

The rotation plane can be defined by 3 points (@P6@P14) in the current tool CS. The target plane is defined when these parameters are transferred. The resulting machining coordinate system is then calculated so that all 3 points lie on the XY plane. The following also applies:

- The X axis of the resulting MCS is parallel to the vector of the 1st reference point in relation to the 2nd reference point
- The Z axis lie perpendicular to the XY plane and is positioned so that its value in Z is positive
- The Y axis is defined so that it forms a right-hand system to the X and Z axes

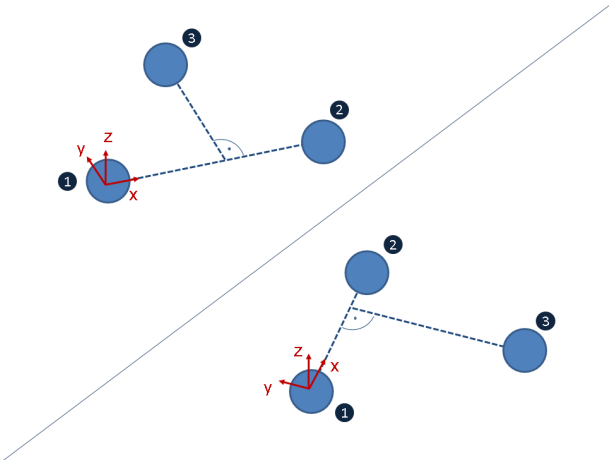

Fig. 2: Definition of the plane

3.2.3 Definition of the plane by angles

The rotation plane can also be defined by angles of rotation (@P15@P17). The target plane is defined when these parameters are transferred. The plane is then implemented by rotation about the axes of the current MCS. Rotation is also possible by projection angle (rotation axes also rotate, @P18=1) and spatial angle (rotation axes remain fixed, @P18=2). The rotation sequence can be defined by @P19.

For more information on rotating an MCS about coordinate system axes, see the documentation on #CS command.

If the rotation cycle is combined with a teach-in cycle to measure inclined planes, the plane can be defined by angles. Here, it is sufficient to transfer the output variables of the teach-in cycle directly: `@P15=V.CYC.SYS_A, @P16=V.CYC.SYS_B, @P17=V.CYC.SYS_C`. The rotation mode and the rotation sequence can remain at the default values.

3.2.4 Parameterisation

The following input parameters are required to call the cycle. The position parameters here refer to the currently active tool coordinate system (before cycle call).

Input parameters	Description
@P1	Rotation mode 1: Machining coordinate system and tool are rotated 2: Tool is rotated 3: Machining coordinate system is rotated 4: No movement, only calculation and output (only practical with @P20=1)
@P2	Retraction distance (relative)
@P3	Position in X of the new origin
@P4	Position in Y of the new origin
@P5	Position in Z of the new origin
@P6	Position in X of the first reference point of the target plane
@P7	Position in Y of the first reference point of the target plane
@P8	Position in Z of the first reference point of the target plane
@P9	Position in X of the second reference point of the target plane
@P10	Position in Y of the second reference point of the target plane
@P11	Position in Z of the second reference point of the target plane
@P12	Position in X of the third reference point of the target plane
@P13	Position in Y of the third reference point of the target plane
@P14	Position in Z of the third reference point of the target plane
@P15	Rotation angle for rotation about the X axis (in degrees)
@P16	Rotation angle for rotation about the Y axis (in degrees)
@P17	Rotation angle for rotation about the Z axis (in degrees)
@P18 (optional)	Rotation mode 1: Intrinsic / projection angle (default) 2: Extrinsic / spatial angle
@P19 (optional)	Rotation sequence (Tait-Bryan angles) 1: XYZ 2: XZY 3: YXZ 4: YZX 5: ZXY 6: ZYX (default)
@P20 (optional)	Output result 0: No output (default 1: Result output via SysPosRotationToPlaneResult.txt

It is recommended using the Syntax check.to verify whether the input parameters have been correctly assigned.

3.2.5 Syntax

L CYCLE [NAME = SysPosRotationToPlane.ecy @P.. = ..]

3.2.6 Programming examples

Example 1 - Plane defined by reference points

Programing Example

Plane defined by reference points

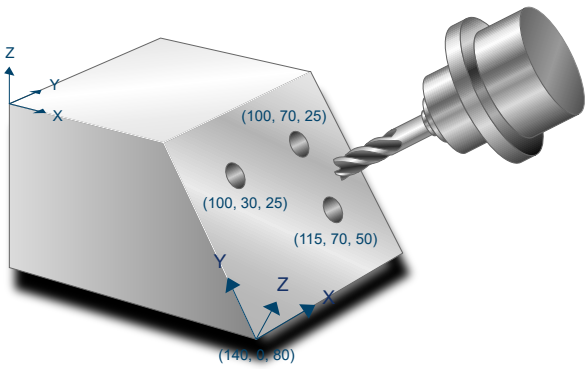

Fig. 3: SysPosRotationToPlaneEx1

The following parameters are used in this example:

- Tool and machining coordinate system are rotated: `@P1=1`
- The retraction distance along the Z axis is 50mm: `@P2 = 50`
- The new origin is located at (140, 0, 80): `@P3=140 @P4=0 @P5=80`
- The first reference point is located at (100, 30, 25): `@P6=100 @P7=30 @P8=25`
- The second reference point is located at (100, 70, 25): `@P9=100 @P10=70 @P11=25`
- The third reference point is located at (115, 70, 50): `@P12=115 @P13=70 @P14=50`

```
#TRAFO ON

L CYCLE [NAME=SysPosRotationToPlane.ecy \
        @P1  = 1                        \
        @P2  = 50                       \
        @P3  = 140                      \
        @P4  = 0                        \
        @P5  = 80                       \
        @P6  = 100                      \
        @P7  = 30                       \
        @P8  = 25                       \
        @P9  = 100                      \
        @P10 = 70                       \
        @P11 = 25                       \
        @P12 = 115                      \
```

```
        @P13 = 70                       \
        @P14 = 50                       \
        ]

(---------------------------------)
(---- Processing the plane  -----)
(---------------------------------)

#CS OFF ; Deactivate the calculated MCS

#TRAFO OFF
```

Example 2 - Plane defined by rotation angles

Programing Example

Plane defined by rotation angles

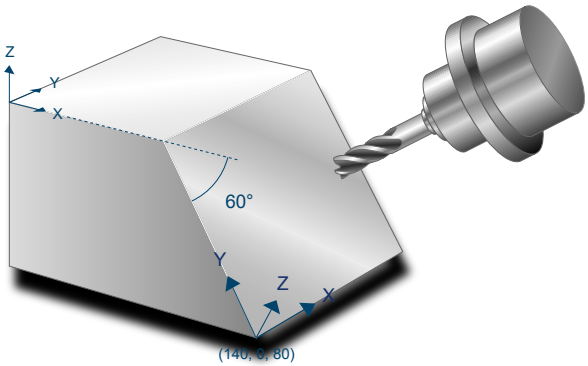

Fig. 4: SysPosRotationToPlaneEx2

The following parameters are used in this example:

- Tool and machining coordinate system are rotated: `@P1=1`
- The retraction distance along the Z axis is 50mm: `@P2 = 50`
- The new origin is located at (140, 0, 80): `@P3=140 @P4=0 @P5=80`
- The first rotation about the Z axis is 90°: `@P17=90`
- The second rotation about the Y axis is 0°: `@P16=0`
- The third rotation about the X axis is -60°: `@P15=-60`

```
#TRAFO ON

L CYCLE [NAME=SysPosRotationToPlane.ecy \
        @P1  = 1                        \
        @P2  = 50                       \
        @P3  = 0                        \
        @P4  = 0                        \
        @P5  = 0                        \
        @P15 = -60                      \
        @P16 = 0                        \
        @P17 = 90                       \
        ]
(---------------------------------)
(---- Processing the plane  -----)
```

```
(--------------------------------)

#CS OFF ; Deactivate the calculated MCS

#TRAFO OFF

M30
```

3.3 SysPosPattern - Generate call pattern

Notice

This function is available as of CNC Build V3.01.3079.37.

The pattern cycle describes a higher-level functionality that allows cycles to be executed multiple times at predefined positions. As opposed to the existing MODAL_MOVE / MODAL_BLOCK function, rotations can also be defined. During cycle execution, they are active in the form of a machining coordinate system.

The list below shows the possible applications that are conceivable in conjunction with the pattern cycle:

- Execute ISG cycles at several predefined positions
- Execute user-defined cycles at several predefined positions
- Define additional rotations which are active at cycle execution

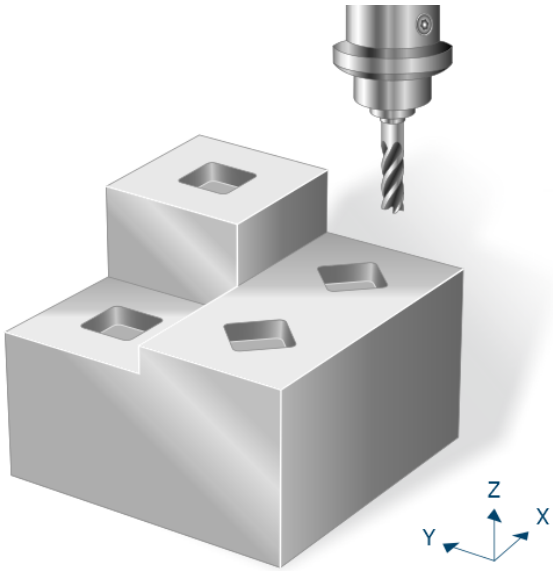

Fig. 5: General call patterns

3.3.1 Define a call pattern in NC program code

Time of definition

No call pattern are predefined when the controller starts up. A definition in the configuration lists is not possible.

Call patterns are defined directly in the NC program in a sequence of position and rotation definitions embedded in plain text commands. The call patterns used must then be defined before a machining cycle is called. The call pattern definition is valid until it is overwritten, deleted or until program end.

Start of a call pattern definition

```
# PATTERN BEGIN [ID<expr>]
```

ID <expr> Identification number of the call pattern.

A call pattern definition is activated by `#PATTERN BEGIN [ID<expr>]`. The freely selectable identification number is then transferred. If a call pattern already exists with the required identification number, it is overwritten by the new call pattern.

End of a call pattern definition

```
# PATTERN END
```

Each call pattern definition must be closed by the command `#PATTERN END`. Standard commands can only be used again when the call pattern definition is closed.

Programming description - Description of a call pattern

```
#PATTERN BEGIN [ID=1]
#AT [100,  0,  0]
#AT [100,100,  0,0,0,90]
#AT [0  ,100,100]
#PATTERN END
```

Each call pattern definition starts with the command `#PATTERN BEGIN` and must be closed by the command `#PATTERN END`. Between these commands, you can describe the required positions and rotations.

Positions and rotations are defined by the command `#AT`. The first 3 input values describe the positions at which the cycle is to be executed. The last 3 input values describe the rotations that are to be activated when the cycle is executed. The angle definition must be selected according to the rotation of a machining coordinate system]. Currently, only one rotation is permitted about the Z axis.

If no rotations are defined, it is sufficient to only define the first 3 values for the `#AT` command.

Delete call pattern

The command to delete a call pattern contains the following syntax elements:

```
# PATTERN DELETE [ID<expr>]
```

ID <expr> Identification number of the call pattern

It is also possible to delete currently defined call patterns and clear the memory location for new definitions.

```
# PATTERN DELETE ALL
```

Information on deleted call patterns is irrevocably lost. The assigned memory location is then cleared for new call pattern definitions. Only existing call pattern definitions can be deleted.

Programming example - Delete call pattern

```
# PATTERN DELETE [ID3] ( Delete specific call pattern with ID3)
# PATTERN DELETE ALL   ( Delete all call patterns)
```

3.3.2 Process

After a call pattern is defined [▸ 17], the related cycles can be called to execute it at the predefined positions. A call pattern retraction plane `PATTERN_RETRACT` must also be defined. This is where the tool retracts to every time before it is repositioned.

When the call pattern ID is transferred at cycle call, the cycle "SysPosPatternWrapper.ecy" stored in the cycle packet is called internally. The position in X and Y is then approached on the call pattern retraction plane for each position and rotation definition (`#AT`) contained in this cycle. The rotations and the transferred Z position are then activated as a machining coordinate system before the required cycle is executed.

3.3.3 Parameterisation

Parameterisation mainly consists of the definition of the call pattern [▸ 17].

Additional parameters transferred at cycle call include:

- The associated call pattern ID `PATTERN_ID`
- The call pattern retraction plane `PATTERN_RETRACT` (absolute parameter) to which the tool is retracted for prepositioning between each of the positions

It is recommended using the Syntax check.to verify whether the input parameters have been correctly assigned.

3.3.4 Syntax

```
; pattern data
# PATTERN BEGIN [ID=V.CYC.PatternID]
# AT [ ... ]
...
# PATTERN END

; cycle call
L CYCLE [NAME=...     \
  @P..          = .. \
  PATTERN_ID     = .. \
  PATTERN_RETRACT = .. \
  ]
```

3.3.5 Programming example

Programing Example

Cycle call with call pattern

The example below describes the milling of a rectangular pocket at three different positions using the call pattern call. The second call pattern definition also includes a rotation that rotates the rectangular pocket at this position by 60 degrees about the Z axis.

```
T8 D8                          ( Tool data )
M6                             ( Tool change )

G54 G90 S6000 M03              ( Technology data )

#VAR
  ; parameters for pattern:
  V.CYC.PatternRetract      = 100
  V.CYC.PatternID           = 1

  ; input parameters:
  V.L.SurfacePosition  = 0    ( Z position of workpiece surface )
  V.L.RetractionPlane  = 50   ( Z position of retraction plane )
  V.L.SafetyClearance  = 2    ( relative value of safety clearance in Z)
```

```
  V.L.DepthOfPocket     = 12    ( depth of pocket )
  V.L.MaxIncrementZ     = 4     ( maximum infeed in Z )
  V.L.MaxIncrementXY    = V.G.WZ_AKT.R ( maximum infeed in XY )
  V.L.FeedRateXY        = 6000 ( machining feedrate in XY )
  V.L.FeedRateZ         = 4000 ( plunging feedrate )
  V.L.MachiningMode     = 1     ( machining mode )
  V.L.PocketLength      = 20    ( length of the pocket )
  V.L.PocketWidth       = 20    ( width of the pocket )
#ENDVAR

; pattern data
#PATTERN BEGIN [ID=V.CYC.PatternID]
#AT [100,   0,  0]
#AT [100, 100,  5, 0, 0, 60]
#AT [0  , 100, 10]
#PATTERN END

; polynomial contouring for smooth movements
#CONTOUR MODE [DEV, PATH_DEV = V.G.WZ_AKT.R / 100]
G261

G00 Z60
G00 X50 Y50                   ( Positioning to the starting point )

L CYCLE [NAME=SysMillRectangularPocket.ecy  \
    @P1  = V.L.SurfacePosition              \
    @P2  = V.L.RetractionPlane              \
    @P3  = V.L.SafetyClearance              \
    @P4  = V.L.DepthOfPocket                \
    @P5  = V.L.MaxIncrementZ                \
    @P6  = V.L.MaxIncrementXY               \
    @P20 = V.L.FeedRateXY                   \
    @P21 = V.L.FeedRateZ                    \
    @P31 = V.L.MachiningMode                \
    @P72 = V.L.PocketLength                 \
    @P73 = V.L.PocketWidth                  \
    PATTERN_ID      = V.CYC.PatternID       \
    PATTERN_RETRACT = V.CYC.PatternRetract  \
    ]

G260
M05
M30
```

4 Calculation cycles

4.1 Introduction

Notice

Cycles are additional options and subject to the purchase of a license.

Task

Calculation cycles are used to execute complex calculations that require the user to have a lot of technical knowledge and carry out extensive programming work.

The following calculation cycles are currently available:

- Calculation of the diameter and centre point of a 2-dimensional circle consisting of min. 3 sampling points
- Calculation of the diameter and centre point of a sphere consisting of min. 4 sampling points
- Calculation of a plane consisting of min. 3 sampling points
- Calculation of a circle in 3D space consisting of min. 3 sampling points
- Calculation of a straight line in 3D space consisting of min. 2 sampling points

The calculation cycles use the “method of least squares” to calculate the most probable result from a specific number of sampling points.

Programming and parameterisation

The programming and parameterisation of cycles is described in each of the overviews in the main chapter.

Handling output variables

If the V.CYC output variable is created in the main program (or in the subroutine calling the cycle), the value is written within the cycle and is available in the main program after the calculation.

The calculation cycles described below include programming examples that can be used to understand how to interpret the output variables.

4.2 Cycle for calculation of a circle in 2D

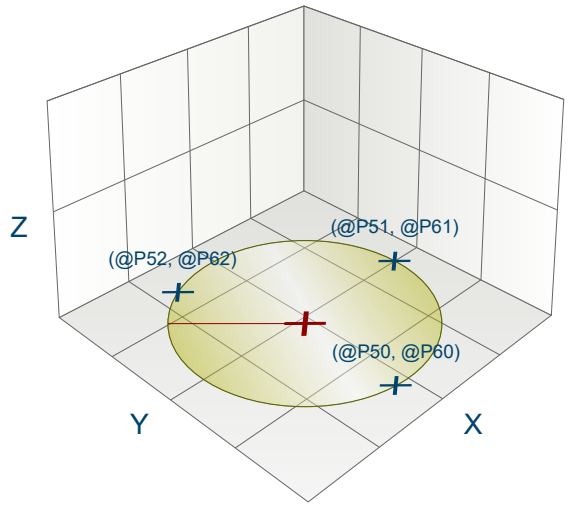

Fig. 6: Calculation of a circle in 2D

4.2.1 Process

When a circle is calculated in the XY plane, the radius and centre of the circle are determined from a specific number of points on the arc (min. 3, max. 10). The points may not be collinear, otherwise no calculation can be performed.

To obtain good calculation results, it is recommended to use points that are as distant from each other as possible.

4.2.2 Parameters

The following parameters are required to call the cycle:

Cycle parameters	Description
@P50	X coordinate point 1
@P51	X coordinate point 2
@P52	X coordinate point 3
@P53 (optional)	X coordinate point 4
@P54 (optional)	X coordinate point 5
@P55 (optional)	X coordinate point 6
@P56 (optional)	X coordinate point 7
@P57 (optional)	X coordinate point 8
@P58 (optional)	X coordinate point 9
@P59 (optional)	X coordinate point 10
@P60	Y coordinate point 1
@P61	Y coordinate point 2
@P62	Y coordinate point 3
@P63 (optional)	Y coordinate point 4

@P64 (optional)	Y coordinate point 5
@P65 (optional)	Y coordinate point 6
@P66 (optional)	Y coordinate point 7
@P67 (optional)	Y coordinate point 8
@P68 (optional)	Y coordinate point 9
@P69 (optional)	Y coordinate point 10

It is recommended using the Syntax check.to verify whether the input parameters have been correctly assigned.

4.2.3 Syntax

L CYCLE [NAME = SysCalcCircle.ecy @P.. = ..]

4.2.4 Output variables

Variable	Value
V.CYC.SysRetCenterX	Calculated centre of circle in X
V.CYC.SysRetCenterY	Calculated centre of circle in Y
V.CYC.SysRetRadius	Calculated radius of circle
V.CYC.SysRetVariance	Calculated variance of the distance of the transferred points from the calculated centre of the circle

See information on use of the output variables [▸ 21].

4.2.5 Programming example

Programing Example

Calculation cycle

```
; Definition of return variables
#VAR
  V.CYC.SysRetRadius
  V.CYC.SysRetCenterX
  V.CYC.SysRetCenterY
  V.CYC.SysRetVariance
#ENDVAR

; cycle call
L CYCLE [NAME=SysCalcCircle.ecy    \
@P50 = 0  @P60 = -1                \
@P51 = 0  @P61 =  1                \
@P52 = 1  @P62 = 0                 \
@P53 = -1  @P63 = 0 ]

; print result
#FILE NAME [MSG="SysCalcCircleResult.txt"]
```

```
#MSG SAVE EXCLUSIVE["Center X = %f",V.CYC.SysRetCenterX]
#MSG SAVE EXCLUSIVE["Center Y = %f",V.CYC.SysRetCenterY]
#MSG SAVE EXCLUSIVE["Radius   = %f", V.CYC.SysRetRadius]
#MSG SAVE EXCLUSIVE["Variance = %f", V.CYC.SysRetVariance]

M30
```

4.3 Cycle for calculation of a plane

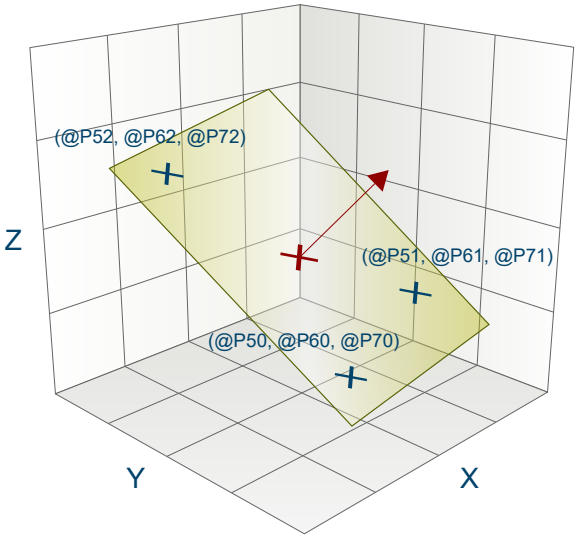

Fig. 7: Calculation of a plane

4.3.1 Process

When a plane is calculated in 3-dimensional space, the normal and support vectors of a plane are determined from a specific number of points (min. 3, max. 10). The points may not be collinear, otherwise no calculation can be performed.

To obtain good calculation results, it is recommended to use points that are as distant from each other as possible.

4.3.2 Parameters

The following parameters are required to call the cycle:

Cycle parameters	Description
@P50	X coordinate point 1
@P51	X coordinate point 2
@P52	X coordinate point 3
@P53 (optional)	X coordinate point 4
@P54 (optional)	X coordinate point 5
@P55 (optional)	X coordinate point 6
@P56 (optional)	X coordinate point 7
@P57 (optional)	X coordinate point 8

@P58 (optional)	X coordinate point 9
@P59 (optional)	X coordinate point 10
@P60	Y coordinate point 1
@P61	Y coordinate point 2
@P62	Y coordinate point 3
@P63 (optional)	Y coordinate point 4
@P64 (optional)	Y coordinate point 5
@P65 (optional)	Y coordinate point 6
@P66 (optional)	Y coordinate point 7
@P67 (optional)	Y coordinate point 8
@P68 (optional)	Y coordinate point 9
@P69 (optional)	Y coordinate point 10
@P70	Z coordinate point 1
@P71	Z coordinate point 2
@P72	Z coordinate point 3
@P73 (optional)	Z coordinate point 4
@P74 (optional)	Z coordinate point 5
@P75 (optional)	Z coordinate point 6
@P76 (optional)	Z coordinate point 7
@P77 (optional)	Z coordinate point 8
@P78 (optional)	Z coordinate point 9
@P79 (optional)	Z coordinate point 10

It is recommended using the Syntax check.to verify whether the input parameters have been correctly assigned.

4.3.3 Syntax

L CYCLE [NAME = SysCalcPlane.ecy @P.. = ..]

4.3.4 Output variables

Variable	Value
V.CYC.SysRetNormalX	Calculated normal vector in X
V.CYC.SysRetNormalY	Calculated normal vector in Y
V.CYC.SysRetNormalZ	Calculated normal vector in Z
V.CYC.SysRetSupVectorX	Calculated support vector of plane in X (corresponds to point on the plane)
V.CYC.SysRetSupVectorY	Calculated support vector of plane in Y corresponds to point on the plane)

V.CYC.SysRetSupVectorZ	Calculated support vector of plane in Z (corresponds to point on the plane)

See information on use of the output variables [▸ 21].

4.3.5 Programming example

Programing Example

Calculation cycle

```
; creation of return variables
#VAR
  V.CYC.SysRetNormalX
  V.CYC.SysRetNormalY
  V.CYC.SysRetNormalZ
  V.CYC.SysRetSupVectorX
  V.CYC.SysRetSupVectorY
  V.CYC.SysRetSupVectorZ
#ENDVAR

; calculation of the XY plane
L CYCLE [NAME=SysCalcPlane.ecy \
@P50 = 1  @P60 = 0  @P70 = 0   \
@P51 = 0  @P61 = 1  @P71 = 0   \
@P52 = 1  @P62 = 1  @P72 = 0   ]

; print result
#FILE NAME[MSG="SysCalcPlaneResult.txt"            ]
#MSG SAVE ["Normal X = %f", V.CYC.SysRetNormalX    ]
#MSG SAVE ["Normal Y = %f", V.CYC.SysRetNormalY    ]
#MSG SAVE ["Normal Z = %f", V.CYC.SysRetNormalZ    ]
#MSG SAVE ["Point X  = %f", V.CYC.SysRetSupVectorX ]
#MSG SAVE ["Point Y  = %f", V.CYC.SysRetSupVectorY ]
#MSG SAVE ["Point Z  = %f", V.CYC.SysRetSupVectorZ ]

M30
```

4.4 Cycle for calculation of a straight line in 3D

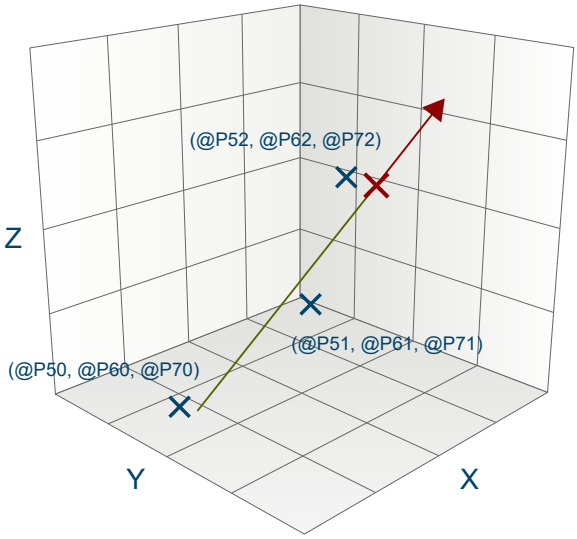

Fig. 8: Calculation of a straight line in 3D

4.4.1 Process

When a straight line is calculated in 3-dimensional space, the distances between the points as well as the direction and support vectors are determined from a specific number of points on the straight line (min. 2, max. 10). The points may not be identical, otherwise no calculation can be performed.

To obtain good calculation results, it is recommended to use points that are as distant from each other as possible.

4.4.2 Parameters

The following parameters are required to call the cycle:

Cycle parameters	Description
@P50	X coordinate point 1
@P51	X coordinate point 2
@P52 (optional)	X coordinate point 3
@P53 (optional)	X coordinate point 4
@P54 (optional)	X coordinate point 5
@P55 (optional)	X coordinate point 6
@P56 (optional)	X coordinate point 7
@P57 (optional)	X coordinate point 8
@P58 (optional)	X coordinate point 9
@P59 (optional)	X coordinate point 10
@P60	Y coordinate point 1
@P61	Y coordinate point 2
@P62 (optional)	Y coordinate point 3

@P63 (optional)	Y coordinate point 4
@P64 (optional)	Y coordinate point 5
@P65 (optional)	Y coordinate point 6
@P66 (optional)	Y coordinate point 7
@P67 (optional)	Y coordinate point 8
@P68 (optional)	Y coordinate point 9
@P69 (optional)	Y coordinate point 10
@P70	Z coordinate point 1
@P71	Z coordinate point 2
@P72 (optional)	Z coordinate point 3
@P73 (optional)	Z coordinate point 4
@P74 (optional)	Z coordinate point 5
@P75 (optional)	Z coordinate point 6
@P76 (optional)	Z coordinate point 7
@P77 (optional)	Z coordinate point 8
@P78 (optional)	Z coordinate point 9
@P79 (optional)	Z coordinate point 10

It is recommended using the Syntax check.to verify whether the input parameters have been correctly assigned.

4.4.3 Syntax

L CYCLE [NAME = SysCalcLine.ecy @P.. = ..]

4.4.4 Output variables

Variable	Value
V.CYC.SysRetDirVectorX	Calculated direction vector of straight line in X
V.CYC.SysRetDirVectorY	Calculated direction vector of straight line in Y
V.CYC.SysRetDirVectorZ	Calculated direction vector of straight line in Z
V.CYC.SysRetSupVectorX	Calculated support vector of straight line in X (corresponds to point on the straight line)
V.CYC.SysRetSupVectorY	Calculated support vector of straight line in Y (corresponds to point on the straight line)
V.CYC.SysRetSupVectorZ	Calculated support vector of straight line in Z (corresponds to point on the straight line)
V.CYC.SysRetVariance	Calculated variance of the distance of the points from the calculated straight line

See information on use of the output variables [▸ 21].

4.4.5 Programming example

Programing Example

Calculation cycle

```
; creation of return variables
#VAR
  V.CYC.SysRetDirVectorX
  V.CYC.SysRetDirVectorY
  V.CYC.SysRetDirVectorZ
  V.CYC.SysRetSupVectorX
  V.CYC.SysRetSupVectorY
  V.CYC.SysRetSupVectorZ
  V.CYC.SysRetVariance
#ENDVAR

; calc line of X-axis from points
L CYCLE [NAME=SysCalcLine.ecy  \
@P50 = 0 @P60 = 0 @P70 = 0     \
@P51 = 1 @P61 = 0 @P71 = 0     \
@P52 = 2 @P62 = 0 @P72 = 0     \
]

#FILE NAME[MSG="SysCalcLineResult.txt"                    ]
#MSG SAVE ["Direction X     = %f", V.CYC.SysRetDirVector[0]]
#MSG SAVE ["Direction Y     = %f", V.CYC.SysRetDirVector[1]]
#MSG SAVE ["Direction Z     = %f", V.CYC.SysRetDirVector[2]]
#MSG SAVE ["Point on Line X = %f", V.CYC.SysRetSupVector[0]]
#MSG SAVE ["Point on Line Y = %f", V.CYC.SysRetSupVector[1]]
#MSG SAVE ["Point on Line Z = %f", V.CYC.SysRetSupVector[2]]
#MSG SAVE ["Variance        = %f", V.CYC.SysRetVariance    ]

M30
```

4.5 Cycle for calculation of a sphere

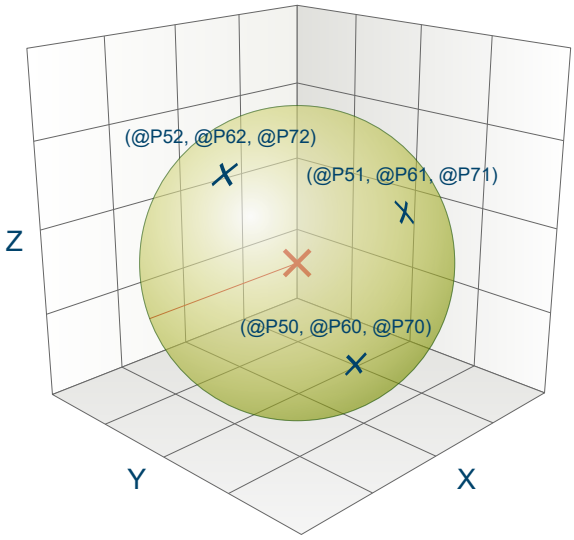

Fig. 9: Calculation of a sphere

4.5.1 Process

When a sphere is calculated in 3-dimensional space, the radius and centre of the sphere are determined from a specific number of points on the sphere's surface (min. 4, max. 10). The points may not be collinear or on the same plane, otherwise no calculation can be performed.

To obtain good calculation results, it is recommended to use points that are as distant from each other as possible.

4.5.2 Parameters

The following parameters are required to call the cycle:

Cycle parameters	Description
@P50	X coordinate point 1
@P51	X coordinate point 2
@P52	X coordinate point 3
@P53	X coordinate point 4
@P54 (optional)	X coordinate point 5
@P55 (optional)	X coordinate point 6
@P56 (optional)	X coordinate point 7
@P57 (optional)	X coordinate point 8
@P58 (optional)	X coordinate point 9
@P59 (optional)	X coordinate point 10
@P60	Y coordinate point 1
@P61	Y coordinate point 2
@P62	Y coordinate point 3

@P63	Y coordinate point 4
@P64 (optional)	Y coordinate point 5
@P65 (optional)	Y coordinate point 6
@P66 (optional)	Y coordinate point 7
@P67 (optional)	Y coordinate point 8
@P68 (optional)	Y coordinate point 9
@P69 (optional)	Y coordinate point 10
@P70	Z coordinate point 1
@P71	Z coordinate point 2
@P72	Z coordinate point 3
@P73	Z coordinate point 4
@P74 (optional)	Z coordinate point 5
@P75 (optional)	Z coordinate point 6
@P76 (optional)	Z coordinate point 7
@P77 (optional)	Z coordinate point 8
@P78 (optional)	Z coordinate point 9
@P79 (optional)	Z coordinate point 10

It is recommended using the Syntax check.to verify whether the input parameters have been correctly assigned.

4.5.3 Syntax

L CYCLE [NAME = SysCalcSphere.ecy @P.. = ..]

4.5.4 Output variables

Variable	Value
V.CYC.SysRetCenterX	Calculated centre of sphere in X
V.CYC.SysRetCenterY	Calculated centre of sphere in Y
V.CYC.SysRetCenterZ	Calculated centre of sphere in Z
V.CYC.SysRetRadius	Calculated radius of sphere
V.CYC.SysRetVariance	Calculated variance of the distance of the transferred points from the calculated sphere

See information on use of the output variables [▸ 21].

4.5.5 Programming example

Programing Example

Calculation cycle

```
; creation of return variables
#VAR
  V.CYC.SysRetCenterX
  V.CYC.SysRetCenterY
  V.CYC.SysRetCenterZ
  V.CYC.SysRetRadius
  V.CYC.SysRetVariance
#ENDVAR

; calculation of the unit sphere
L CYCLE [NAME=SysCalcSphere.ecy   \
@P50 =  1 @P60 =  0  @P70 = 0     \
@P51 = -1 @P61 =  0  @P71 = 0     \
@P52 =  0 @P62 =  1  @P72 = 0     \
@P53 =  0 @P63 = -1  @P73 = 0     \
@P54 =  0 @P64 =  0  @P74 = 1     \
]

; print result
#FILE NAME[MSG="SysCalcSphereResult.txt"]
#MSG SAVE EXCLUSIVE["Center X = %f", V.CYC.SysRetCenterX ]
#MSG SAVE EXCLUSIVE["Center Y = %f", V.CYC.SysRetCenterY ]
#MSG SAVE EXCLUSIVE["Center Z = %f", V.CYC.SysRetCenterZ ]
#MSG SAVE EXCLUSIVE["Radius   = %f", V.CYC.SysRetRadius  ]
#MSG SAVE EXCLUSIVE["Variance = %f", V.CYC.SysRetVariance]

M30
```

4.6 Cycle for calculation of a circle in 3D

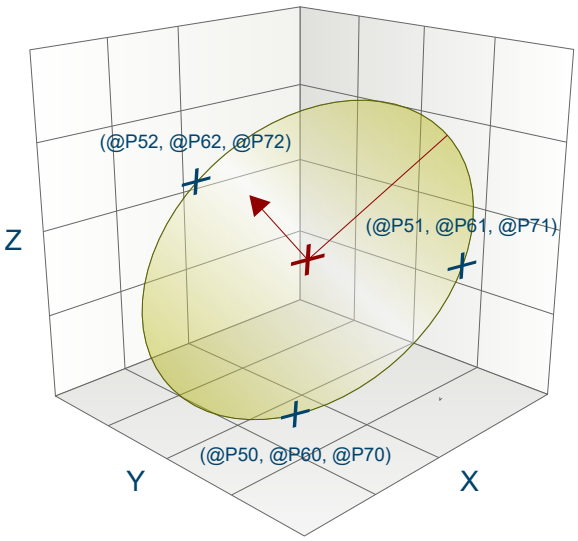

Fig. 10: Calculation of a circle in 3D

4.6.1 Process

When a circle is calculated in 3-dimensional space, the radius, distance from origin and normal vector of the circle are determined from a specific number of points on the arc (min. 3, max. 10). The points may not be collinear, otherwise no calculation can be performed.

To obtain good calculation results, it is recommended to use points that are as distant from each other as possible.

4.6.2 Parameters

The following parameters are required to call the cycle:

Cycle parameters	Description
@P50	X coordinate point 1
@P51	X coordinate point 2
@P52	X coordinate point 3
@P53 (optional)	X coordinate point 4
@P54 (optional)	X coordinate point 5
@P55 (optional)	X coordinate point 6
@P56 (optional)	X coordinate point 7
@P57 (optional)	X coordinate point 8
@P58 (optional)	X coordinate point 9
@P59 (optional)	X coordinate point 10
@P60	Y coordinate point 1
@P61	Y coordinate point 2
@P62	Y coordinate point 3
@P63 (optional)	Y coordinate point 4
@P64 (optional)	Y coordinate point 5
@P65 (optional)	Y coordinate point 6
@P66 (optional)	Y coordinate point 7
@P67 (optional)	Y coordinate point 8
@P68 (optional)	Y coordinate point 9
@P69 (optional)	Y coordinate point 10
@P70	Z coordinate point 1
@P71	Z coordinate point 2
@P72	Z coordinate point 3
@P73 (optional)	Z coordinate point 4
@P74 (optional)	Z coordinate point 5
@P75 (optional)	Z coordinate point 6
@P76 (optional)	Z coordinate point 7

@P77 (optional)	Z coordinate point 8
@P78 (optional)	Z coordinate point 9
@P79 (optional)	Z coordinate point 10

It is recommended using the Syntax check.to verify whether the input parameters have been correctly assigned.

4.6.3 Syntax

L CYCLE [NAME = SysCalcPlaneCircle.ecy @P.. = ..]

4.6.4 Output variables

Variable	Value
V.CYC.SysRetCenterX	Calculated centre of circle in X
V.CYC.SysRetCenterY	Calculated centre of circle in Y
V.CYC.SysRetCenterZ	Calculated centre of circle in Z
V.CYC.SysRetRadius	Calculated radius of circle
V.CYC.SysRetNormalX	Calculated normal vector in X
V.CYC.SysRetNormalY	Calculated normal vector in Y
V.CYC.SysRetNormalZ	Calculated normal vector in Z
V.CYC.SysRetRadVariance	Calculated variance of points to radius

See information on use of the output variables [▸ 21].

4.6.5 Programming example

Programing Example

Calculation cycle

```
; creation of return variables
#VAR
  V.CYC.SysRetCenterX
  V.CYC.SysRetCenterY
  V.CYC.SysRetCenterZ
  V.CYC.SysRetNormalX
  V.CYC.SysRetNormalY
  V.CYC.SysRetNormalZ
  V.CYC.SysRetRadius
  V.CYC.SysRetRadVariance
#ENDVAR

; calculation of unit circle in the XY-plane with Z=1
L CYCLE [NAME=SysCalcPlaneCircle.ecy  \
@P50 =  1  @P60 =  0  @P70 = 1        \
@P51 =  0  @P61 =  1  @P71 = 1        \
@P52 = -1  @P62 =  0  @P72 = 1        \
```

```
@P53 =  0  @P63 = -1  @P73 = 1        ]

; print result
#FILE NAME[MSG="SysCalcPlaneCircleResult.txt"        ]
#MSG SAVE ["Radius   = %f", V.CYC.SysRetRadius       ]
#MSG SAVE ["Center X = %f", V.CYC.SysRetCenterX      ]
#MSG SAVE ["Center Y = %f", V.CYC.SysRetCenterY      ]
#MSG SAVE ["Center Z = %f", V.CYC.SysRetCenterZ      ]
#MSG SAVE ["Normal X = %f", V.CYC.SysRetNormalX      ]
#MSG SAVE ["Normal Y = %f", V.CYC.SysRetNormalY      ]
#MSG SAVE ["Normal Z = %f", V.CYC.SysRetNormalZ      ]
#MSG SAVE ["Variance = %f", V.CYC.SysRetRadVariance  ]

M30
```

5 High Speed Settings

5.1 Introduction

In general, the machine response can be optimised by means of channel parameters using the High Speed Settings cycle. This allows you to influence the surface quality, speed and accuracy for a specific machine.

Notice

Cycles are additional options and subject to the purchase of a license.

Task

The High Speed Setting cycles optimise the machining response of a specific machine by specifying tolerances for

- roughing
- prefinishing
- finishing

. The user can use the cycle to select the required machining state to influence machine response.

Licensing note

Please note that cycles are additional options and subject to the purchase of a license.

5.2 SysHscSettings cycle - High Speed Cutting settings

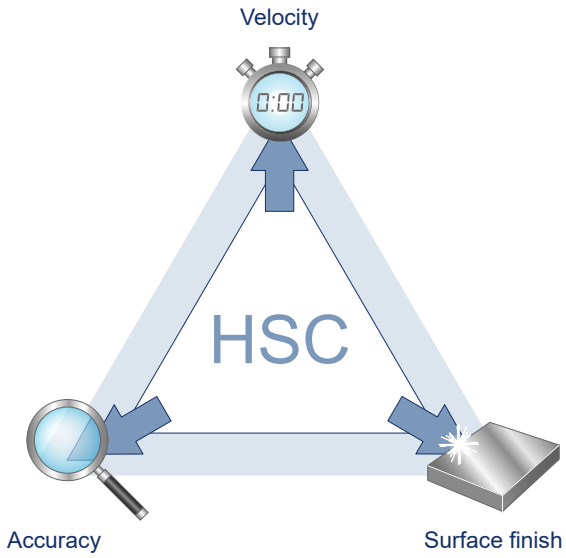

Fig. 11: Operation mode of HSC settings

5.2.1 Process

The SysHscSettings cycle can optimise machine response. NC programs can be specified in the channel-specific lists. They are called depending on the specified mode.

The parameters for this are:

- hscs.deselect.prog
- hscs.deselect.tolerance
- hscs.rough.prog
- hscs.rough.tolerance
- hscs.prefinish.prog
- hscs.prefinish.tolerance
- hscs.finish.prog
- hscs.finish.tolerance

The values for the tolerance data lists are converted internally into mm or inch.

These tolerances can be read in the NC program by the following variables:

```
V.G.HSCS.DESELECT.TOLERANCE
V.G.HSCS.ROUGH.TOLERANCE
V.G.HSCS.PREFINISH.TOLERANCE
V.G.HSCS.FINISH.TOLERANCE
```

5.2.2 Parameterisation

The following input parameters are required to call the cycle:

Input parameters	Description
@P1	Mode default value = 1 1 = Deselect 2 = Rough (roughing) 3 = Medium (prefinishing) 4 = Fine (finishing)

It is recommended using the Syntax check.to verify whether the input parameters have been correctly assigned.

5.2.3 Syntax

L CYCLE [NAME = SysHscSettings.ecy @P.. = ..]

5.2.4 Programming example

General programming example

Programing Example

Mill circular pocket

Mode 4 is called in this example. This mode corresponds to the finishing configuration. As a consequence, the subroutine manufacturerHscFinishSettings.nc specified in the parameter hscs.finish.prg is called. In addition, this makes available the machining tolerance of 0.01 mm in the variable V.G.HSCS.FINISH.TOLERANCE and it is used in the subroutine as an example for the `#HSC` command.

```
; HSC Settings
T9 D9                                    ( Tool data )
```

```
M6                                          ( Tool change )
G00 G17 G90 F2000 M03 S6000                 ( Technology data )
G00 Z150                                    ( Go to z start position )
G00 X0 Y0                                   ( position over the workpiece )

; Finishing Mode
L CYCLE [NAME=SysHscSettings.ecy    \
        @P1 = 4                     \
        ]


; 3D Milling
;...
;...

M30
```

Programming example for optimisation programs

The specified data are given as an example and cannot simply be adopted. This could cause damage to the machine.

```
% manufacturerHscDeselectSettings.nc
G133 = 100
G134 = 100
#HSC OFF
#FILTER OFF
M17


% manufacturerHscRoughSettings.nc
; example
; G133 = 180 ; example G133 = 180 set 180%
; G134 = 180
; #SLOPE[TYPE=HSC]
; #HSC ON[SURFACE PATH_DEV=V.G.HSCS.ROUGH.TOLERANCE \
                  TRACK_DEV=V.G.HSCS.ROUGH.TOLERANCE]
; #FILTER ON [AX_DEV=V.G.HSCS.ROUGH.TOLERANCE]
M17


% manufacturerHscPrefinishSettings.nc
; example
; G133 = 150 ; example G133 = 150 set 150%
; G134 = 150
; #SLOPE[TYPE=HSC]
; #HSC ON[SURFACE PATH_DEV=V.G.HSCS.PREFINISH.TOLERANCE \
                  TRACK_DEV=V.G.HSCS.PREFINISH.TOLERANCE]
; #FILTER ON [AX_DEV=V.G.HSCS.PREFINISH.TOLERANCE]
M17


% manufacturerHscFinishSettings.nc
; example
; G133 = 80 ; example G133 = 80 set 80%
; G134 = 80
; #SLOPE[TYPE=HSC]
; #HSC ON[SURFACE PATH_DEV=V.G.HSCS.FINISH.TOLERANCE \
           TRACK_DEV=V.G.HSCS.FINISH.TOLERANCE]
; #FILTER ON [AX_DEV=V.G.HSCS.FINISH.TOLERANCE]
M17
```

6 Appendix

6.1 Suggestions, corrections and the latest documentation

Did you find any errors? Do you have any suggestions or constructive criticism? Then please contact us at documentation@isg-stuttgart.de. The latest documentation is posted in our Online Help (DE/EN):

QR code link: https://www.isg-stuttgart.de/documentation-kernel/

The link above forwards you to:

https://www.isg-stuttgart.de/fileadmin/kernel/kernel-html/index.html

Notice

Change options for favourite links in your browser;

Technical changes to the website layout concerning folder paths or a change in the HTML framework and therefore the link structure cannot be excluded.

We recommend you to save the above "QR code link" as your primary favourite link.

PDFs for download:

DE:
https://www.isg-stuttgart.de/produkte/softwareprodukte/isg-kernel/dokumente-und-downloads

EN:
https://www.isg-stuttgart.de/en/products/softwareproducts/isg-kernel/documents-and-downloads

E-Mail: documentation@isg-stuttgart.de

www.isg-stuttgart.de
support@isg-stuttgart.de