

# Module Interface Specification for STEM Moiré GPA

Alexandre Pofelski  
macid: pofelska  
github: slimpotatoes

November 21, 2017

# 1 Revision History

Date	Version	Notes
14/11/2017	1.0	First draft

## 2 Symbols, Abbreviations and Acronyms

The same Symbols, Abbreviations and Acronyms as in the SRS, the TestPlan and the MG (available in [STEM Moiré GPA](#) repository) are used in the Module Interface Specifications document.

addition to document

# Contents

<b>1</b>	<b>Revision History</b>	<b>i</b>
<b>2</b>	<b>Symbols, Abbreviations and Acronyms</b>	<b>ii</b>
<b>3</b>	<b>Introduction</b>	<b>1</b>
<b>4</b>	<b>Notation</b>	<b>1</b>
<b>5</b>	<b>Module Decomposition</b>	<b>1</b>
<b>6</b>	<b>MIS of STEM Moiré GPA Control Module (M 2)</b>	<b>3</b>
6.1	Module . . . . .	3
6.2	Uses . . . . .	3
6.3	Syntax . . . . .	3
6.3.1	Exported Access Programs . . . . .	3
6.4	Semantics . . . . .	3
6.4.1	State Variables . . . . .	3
6.4.2	Access Routine Semantics . . . . .	3
<b>7</b>	<b>MIS of STEM Moiré GPA GUI Module (M 3)</b>	<b>4</b>
7.1	Module . . . . .	4
7.2	Uses . . . . .	4
7.3	Syntax . . . . .	4
7.3.1	Exported Access Programs . . . . .	4
7.4	Semantics . . . . .	4
7.4.1	State Variables . . . . .	5
7.4.2	Access Routine Semantics . . . . .	5
<b>8</b>	<b>MIS of Input Module (M 4)</b>	<b>7</b>
8.1	Module . . . . .	7
8.2	Uses . . . . .	7
8.3	Syntax . . . . .	7
8.3.1	Exported Access Programs . . . . .	7
8.4	Semantics . . . . .	7
8.4.1	State Variables . . . . .	7
8.4.2	Access Routine Semantics . . . . .	7
<b>9</b>	<b>MIS of SMH Simulation (M 5)</b>	<b>8</b>
9.1	Module . . . . .	8
9.2	Uses . . . . .	8
9.3	Syntax . . . . .	8
9.3.1	Exported Access Programs . . . . .	8

9.4	Semantics . . . . .	8
9.4.1	State Variables . . . . .	8
9.4.2	Access Routine Semantics . . . . .	8
<b>10</b>	<b>MIS of GPA Module (M 6)</b>	<b>9</b>
10.1	Module . . . . .	9
10.2	Uses . . . . .	9
10.3	Syntax . . . . .	9
10.3.1	Exported Access Programs . . . . .	9
10.4	Semantics . . . . .	9
10.4.1	State Variables . . . . .	9
10.4.2	Access Routine Semantics . . . . .	10
<b>11</b>	<b>MIS of Mask Module (M 7)</b>	<b>10</b>
11.1	Module . . . . .	10
11.2	Uses . . . . .	10
11.3	Syntax . . . . .	10
11.3.1	Exported Access Programs . . . . .	10
11.4	Semantics . . . . .	10
11.4.1	State Variables . . . . .	10
11.4.2	Access Routine Semantics . . . . .	10
<b>12</b>	<b>MIS of Unstrained region (M 8)</b>	<b>11</b>
12.1	Module . . . . .	11
12.2	Uses . . . . .	11
12.3	Syntax . . . . .	11
12.3.1	Exported Access Programs . . . . .	11
12.4	Semantics . . . . .	11
12.4.1	State Variables . . . . .	11
12.4.2	Access Routine Semantics . . . . .	11

### 3 Introduction

The following document details the Module Interface Specifications for STEM Moiré GPA. The full documentation and implementation can be found in [STEM Moiré GPA](#) repository.

### 4 Notation

The structure of the MIS for modules comes from [?], with the addition that template modules have been adapted from [?]. The mathematical notation comes from Chapter 3 of [?]. For instance, the symbol  $:=$  is used for a multiple assignment statement and conditional rules follow the form  $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | \dots | c_n \Rightarrow r_n)$ .

The following table summarizes the primitive data types used by STEM Moiré GPA.

Data Type	Notation	Description
character	char	a single symbol or digit
integer	$\mathbb{Z}$	an integer number
natural number	$\mathbb{N}$	a natural number
real	$\mathbb{R}$	a real number

The specification of STEM Moiré GPA uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, STEM Moiré GPA uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

### 5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	Level 2
Hardware-Hiding Module	
	Input
	STEM Moiré GPA Control
	STEM Moiré GPA GUI
	User Input
Behaviour-Hiding Module	SMH simulation
	GPA
	Mask
	Unstrained region
	Conversion
	2D strain tensor
	Fourier Transform
	Least square fitting method
Software Decision Module	Phase calculation
	Gradient
	Generic GUI/Plot
	Data structure

Table 1: Module Hierarchy

LIST ALL MIS to refer them in other document

## 6 MIS of STEM Moiré GPA Control Module (M 2)

### 6.1 Module

main

### 6.2 Uses

- STEM Moiré GPA GUI
- Processing modules
  - Unstrained region
  - Conversion
  - SMH Simulation
  - GPA
  - 2D Strain Tensors
- Input
- Data Structure

### 6.3 Syntax

#### 6.3.1 Exported Access Programs

Name	In	Out	Exceptions
main	-	-	-

### 6.4 Semantics

STEM Moiré GPA is designed to have the different steps of the process flow driven by user directly through GUI\_SMG. The STEM Moiré GPA Control Module uses the events in STEM Moiré GPA GUI to use the processing modules in the order defined by the user.

#### 6.4.1 State Variables

#### 6.4.2 Access Routine Semantics

main():

- transition: A reflexir
- GUIFlow(): *# Software permanently running until user abort it by closing the GUI*



```

(event_Input()
→ Get the path pathISMH and pathIC from the user ∧
load_files(pathISMH,pathIC))
(event_SimSMH() → SMHsim(load(ISMHexp), load(ICref), load(pISMHexp), load(pICref))
(event_GPA() → gpa(load(FTISMHexp), MCirc(collect_(M))))
(event_URef() → ZeroStrain())
(event_Conversion() → conversion())
(event_StrainCalc() → CalcStrain())

```

## 7 MIS of STEM Moiré GPA GUI Module (M 3)

### 7.1 Module

GUI\_SMG

### 7.2 Uses

- Generic GUI/Plot
- Data Structure

### 7.3 Syntax

#### 7.3.1 Exported Access Programs

Name	In	Out	Exceptions
GUIFlow	-	-	-
GUI_SMHexp	-	-	-
GUI_SMHSim	-	-	-
GUI_Phase	-	-	-
GUI_Conv	-	-	-
event_Input	-	-	-
event_SMHSim	-	-	-
event_GPA	-	-	-
event_URef	-	-	-
event_StrainCalc	-	-	-
collect_M	GUI object	object	-
collect_U	GUI object	object	-

### 7.4 Semantics

STEM Moiré GPA process flow is driven by user through GUI\_SMG. User triggers the events that start the selected processing step.

### 7.4.1 State Variables

Win\_Flow: GUI object  
Win\_SMHexp: GUI object  
Win\_SMHSim: GUI object  
Win\_Phase: GUI object  
Win\_FTSMH: GUI object  
Win\_Conv: GUI object

### 7.4.2 Access Routine Semantics

*# GUI embedding the process flow into buttons triggering events. It is the user role to execute the process flow*

GUIFlow():

- transition:
  1. Win\_Flow=fig('Win\_Flow')
  2. button(Win\_Flow,5,'Input','SMHSim','GPA','URef','StrainCalc')
  3. plot()

*# Events triggered by each button pressed by the user*

event\_Input():

- transition: Trigger event\_Input when button\_Input pressed

event\_SMHSim():

- transition: Trigger event\_SMHSim when button\_SMHSim pressed

event\_GPA():

- transition: Trigger event\_GPA when button\_GPA pressed

event\_URef():

- transition: Trigger event\_URef when button\_URef pressed

event\_StrainCalc():

- transition: Trigger event\_StrainCalc when button\_StrainCalc press

*# GUI to display the display the input files  $I_{SMH_{exp}}$ ,  $I_{C_{ref}}$*

GUI\_SMHexp():

- transition:

1. Win\_SMHexp=fig('Win\_SMHexp',load( $I_{SMH_{exp}}$ ), load( $I_{C_{ref}}$ ))
2. plot()

*# GUI to display the simulation of the STEM Moiré hologram using the reference image and to let the user input  $M$  (from R 4, R 5)*

GUI\_SMHSim():

- transition:

1. Win\_SMHSim=fig('Win\_SMHSim',load( $\tilde{I}_{SMH_{exp}}$ ),load( $\tilde{I}_{SMH_{sim}}$ ),circle( $M$ ))
2. plot()

*# GUI to display the phase resulting from the GPA algorithm and to let the user input  $U$  (from R 8)*

GUI\_Phase():

- transition:

1. Win\_Phase=fig('Win\_Phase',red( $\Delta \vec{g}^{M_{exp}}$ ),rectangle( $U$ ))
2. plot()

*# GUI to display the window to let the user input  $n$  and  $m$  (from R 11)*

GUI\_Conv():

- transition:

1. Win\_Conv=fig('Win\_Conv',entry\_field( $n$ ),entry\_field( $m$ ))
2. plot()

*# Reader of the GUI objects drawn by the user (circle  $M$  or rectangle  $U$ )*

collect\_M( $A$ )

- output:  $C$  such that

1. Verify the type GUI  $A$  to match a circle

2.  $C=(x_c, y_c, R)$  with  $(x_c, y_c)$  the coordinate (pixel number) of the center of the circle  $A$  and  $R$  the radius of the circle  $A$ .

collect\_U( $A$ )

- output:  $r$  such that
  1. Verify the type GUI  $A$  to match a rectangle
  2.  $r=bla$  with  $(x_c, y_c)$

## 8 MIS of Input Module (M 4)

### 8.1 Module

Input

### 8.2 Uses

- STEM Moié GPA GUI
- Data Structure

### 8.3 Syntax

#### 8.3.1 Exported Access Programs

Name	In	Out	Exceptions
load_files	string	-	FilePath

### 8.4 Semantics

#### 8.4.1 State Variables

#### 8.4.2 Access Routine Semantics

load\_files(pathISMH,pathIC):

- transition: pathISMH and pathIC are the file paths for the input files. The following procedure is performed:
  1. The .dm3 files are read and their respective metafiles are collected.
  2. From the metafile,  $I_{SMH_{exp}}$ ,  $I_{C_{ref}}$ ,  $p$  and  $p_{ref}$  are extracted to modify their respective state variable.

3. The variables  $I_{SMH_{\text{exp}}}$ ,  $I_{C_{\text{ref}}}$ ,  $p$  and  $p_{\text{ref}}$  are stored in the data structure:

- store(ISMHexp,  $I_{SMH_{\text{exp}}}$ )
- store(pISMexp,  $p$ )
- store(ICref,  $I_{C_{\text{ref}}}$ )
- store(pICref,  $p_{\text{ref}}$ )

- output:  $I_{SMH_{\text{exp}}}$ ,  $I_{C_{\text{ref}}}$ ,  $p$ ,  $p_{\text{ref}}$
- exception:

## 9 MIS of SMH Simulation (M 5)

### 9.1 Module

SMHSimCalc

### 9.2 Uses

- Fourier Transform
- Data Structure

### 9.3 Syntax

#### 9.3.1 Exported Access Programs

Name	In	Out	Exceptions
SMHsim	$I_{SMH_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{R}$ $I_{C_{\text{ref}}} : \mathbb{R}^2 \rightarrow \mathbb{R}$ , $p \in \mathbb{R}^{+*}$ , $p_{\text{ref}} \in \mathbb{R}^{+*}$	$\tilde{I}_{SMH_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{C}$ $\tilde{I}_{SMH_{\text{sim}}} : \mathbb{R}^2 \rightarrow \mathbb{C}$ $N_{\text{lim}} \in \mathbb{N}^*$	Nlim.zero()

### 9.4 Semantics

#### 9.4.1 State Variables

data : object

#### 9.4.2 Access Routine Semantics

SMHsim( $I_{SMH_{\text{exp}}}$ ,  $I_{C_{\text{ref}}}$ ,  $p$ ,  $p_{\text{ref}}$ ):

- transition:

- store( $\text{FTISMH}_{\text{exp}}, \tilde{I}_{SMH_{\text{exp}}}$ ) such that

$$\tilde{I}_{SMH_{\text{exp}}}(\vec{\nu}) = \mathcal{FT}[I_{SMH_{\text{exp}}}(\vec{r})]$$

- store( $\text{FTISMH}_{\text{sim}}, \tilde{I}_{SMH_{\text{sim}}}$ ) such that

$$\tilde{I}_{SMH_{\text{sim}}}(\vec{\nu}) = \frac{1}{p^2} \sum_{\vec{q} \in Q_{\text{lim}}} \mathcal{FT}[I_{C_{\text{ref}}}(\vec{\nu} - \frac{\vec{q}}{p})]$$

with  $Q_{\text{lim}} = \{\forall(n, m) \in \mathbb{Z}^2 \cap [-N_{\text{lim}}, N_{\text{lim}}]^2, \vec{q} = n\vec{u}_x + m\vec{u}_y\}$

and  $N_{\text{lim}} = \Xi(\frac{p}{p_{\text{ref}}})$  with  $\Xi$  the floor function

- exception:

## 10 MIS of GPA Module (M 6)

### 10.1 Module

GPACalc

### 10.2 Uses

- Mask
- Fourier Transform
- Phase
- Gradient
- Data Structure

### 10.3 Syntax

#### 10.3.1 Exported Access Programs

Name	In	Out	Exceptions
gpa	$\tilde{I}_{SMH_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{C}$ , $M : \mathbb{R}^2 \rightarrow \mathbb{R}$ , $\vec{g}^{M_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{R}$	$P_{\vec{g}} : \mathbb{R}^2 \rightarrow \mathbb{R}$ , $\vec{\Delta g} : \mathbb{R}^2 \rightarrow \mathbb{R}$	-

### 10.4 Semantics

#### 10.4.1 State Variables

data : object

### 10.4.2 Access Routine Semantics

$\text{gpa}(\tilde{I}_{SMH_{\text{exp}}}, M, \vec{g}^{M_{\text{exp}}})$ :

- output:

- $P_{\vec{g}}$  such that

$$\forall \vec{r} \in \mathbb{R}^2, P_{\vec{g}}(\vec{r}) = \arg(i\mathcal{FT}[M \times \tilde{I}_{SMH_{\text{exp}}}] )$$

- $\Delta \vec{g}$  such that

$$\forall \vec{r} \in \mathbb{R}^2, \Delta \vec{g}(\vec{r}) = \frac{1}{2\pi} \text{grad}(\text{unwrap}(P_{\vec{g}}(\vec{r}))) - \vec{g}^{M_{\text{exp}}}(\vec{r})$$

- $P_{\Delta \vec{g}}$  such that

$$\forall \vec{r} \in \mathbb{R}^2, P_{\Delta \vec{g}}(\vec{r}) = \text{wrap}(\text{unwrap}[P_{\vec{g}}(\vec{r})] - 2\pi \vec{g}^{M_{\text{exp}}}(\vec{0}) \cdot \vec{r})$$

- exception:

## 11 MIS of Mask Module (M 7)

### 11.1 Module

Mask

### 11.2 Uses

- STEM Moiré GPA GUI
- Data structure

### 11.3 Syntax

#### 11.3.1 Exported Access Programs

Name	In	Out	Exceptions
MCirc	$(x_c, y_c) \in \mathbb{N}^2, R \in \mathbb{R}^{+*}$	$M : \mathbb{R}^2 \rightarrow \mathbb{R}, \vec{g}_0 : \mathbb{R}^2 \rightarrow \mathbb{R}^2$	-

### 11.4 Semantics

#### 11.4.1 State Variables

#### 11.4.2 Access Routine Semantics

$\text{MCirc}(x_c, y_c, R)$ :

- output:  $(M, \vec{g}_0)$ 
  - $M$  such that

$$M(x, y) = \begin{cases} 1, & (x - x_c)^2 + (y - y_c)^2 \leq R^2 \\ 0, & (x - x_c)^2 + (y - y_c)^2 > R^2 \end{cases}$$

- $\vec{g}_0$  such that

$$\forall \vec{r} \in \mathbb{R}^2, \vec{g}_0(\vec{r}) = \begin{bmatrix} x_c \\ y_c \end{bmatrix}$$

- exception:

## 12 MIS of Unstrained region (M 8)

### 12.1 Module

URefCalc

### 12.2 Uses

- Least Square Fit
- STEM Moiré GPA GUI
- Data Structure

### 12.3 Syntax

#### 12.3.1 Exported Access Programs

Name	In	Out	Exceptions
ZeroStrain	$\vec{\Delta g}^M : \mathbb{R}^2 \rightarrow \mathbb{R}^2, U \in \mathbb{R}^2$	$\vec{\Delta g}_{\text{cor}}^M : \mathbb{R}^2 \rightarrow \mathbb{R}^2$	-

### 12.4 Semantics

#### 12.4.1 State Variables

#### 12.4.2 Access Routine Semantics

ZeroStrain( $\vec{\Delta g}^M, U$ ):

- transition:
- output:  $\vec{\Delta g}_{\text{cor}}^M$  such that

$$\vec{\Delta g}_{\text{cor}}^M = \vec{\Delta g}^M - \text{lsfm}(\vec{\Delta g}^M, U)$$



- exception:

## 13 MIS of Conversion Module (M 9)

### 13.1 Module

MtoCConv

### 13.2 Uses

- Input
- Data Structure

### 13.3 Syntax

#### 13.3.1 Exported Access Programs

Name	In	Out	Exceptions
conversion	$p \in \mathbb{R}$ , $(n, m) \in \mathbb{N}^2$ , $\vec{g}_{\text{uns}}^{M_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$	$\vec{g}_{\text{uns}}^{C_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$	-

### 13.4 Semantics

#### 13.4.1 State Variables

#### 13.4.2 Access Routine Semantics

conversion( $p, \vec{g}_{\text{uns}}^{M_{\text{exp}}}$ ):

- output:  $\vec{g}_{\text{uns}}^{M_{\text{exp}}}$  such that

$$\forall \vec{r} \in \mathbb{R}^2, \vec{g}_{\text{uns}}^{C_{\text{exp}}}(\vec{r}) = \vec{g}_{\text{uns}}^{M_{\text{exp}}}(\vec{r}) + p \times \begin{bmatrix} n \\ m \end{bmatrix}$$

- exception:

## 14 MIS of 2D Strain Tensor Module (M 10)

### 14.1 Module

2D\_Strain

## 14.2 Uses

Data Structure

## 14.3 Syntax

### 14.3.1 Exported Access Programs

Name	In	Out	Exceptions
CalcStrain	$g_{1_{\text{uns}}}^{C_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ , $g_{2_{\text{uns}}}^{C_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ , $\Delta g_{1_{\text{uns}}}^{C_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ , $\Delta g_{2_{\text{uns}}}^{C_{\text{exp}}} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$	$T : \mathbb{R}^2 \rightarrow \mathbb{R}^4$	-

## 14.4 Semantics

### 14.4.1 State Variables

### 14.4.2 Access Routine Semantics

$\text{CalcStrain}(g_{1_{\text{uns}}}^{C_{\text{exp}}}, g_{2_{\text{uns}}}^{C_{\text{exp}}}, \Delta g_{1_{\text{uns}}}^{C_{\text{exp}}}, \Delta g_{2_{\text{uns}}}^{C_{\text{exp}}})$ :

- output:
- exception:

## 15 MIS of Fourier Transform Module (M 11)

*# 2D Fourier transform*

## 15.1 Module

FTCalc

## 15.2 Uses

Data Structure

## 15.3 Syntax

### 15.3.1 Exported Access Programs

Name	In	Out	Exceptions
$\mathcal{FT}$	$f : \mathbb{R}^2 \rightarrow \mathbb{R}$	$f : \mathbb{R}^2 \rightarrow \mathbb{C}$	-
$\text{i}\mathcal{FT}$	$f : \mathbb{R}^2 \rightarrow \mathbb{C}$	$f : \mathbb{R}^2 \rightarrow \mathbb{R}$	-

## 15.4 Semantics

### 15.4.1 State Variables

None

### 15.4.2 Access Routine Semantics

*# Calculate the 2D Fourier transform of a function  $f$*

$\mathcal{FT}(f(x, y))$ :

- output:  $\tilde{f}(\nu, \mu)$  such that

$$\forall(\nu, \mu) \in \mathbb{R}^2 \wedge \forall(x, y) \in \mathbb{R}^2, \tilde{f}(\nu, \mu) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-2i\pi(\nu x + \mu y)} dx dy$$

- exception:

*# Calculate the 2D inverse Fourier transform of a function  $\tilde{f}$*

$\mathcal{iFT}(\tilde{f}(\nu, \mu))$ :

- output:  $f(x, y)$  such that

$$\forall(x, y) \in \mathbb{R}^2 \wedge \forall(\nu, \mu) \in \mathbb{R}^2, f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \tilde{f}(\nu, \mu) e^{2i\pi(\nu x + \mu y)} dx dy$$

- exception:

## 16 MIS of Gradient Module (M 12)

*# 2D Gradient*

### 16.1 Module

GradCalc

### 16.2 Uses

Data Structure

### 16.3 Syntax

#### 16.3.1 Exported Access Programs

Name	In	Out	Exceptions
grad	$f : \mathbb{R}^2 \rightarrow \mathbb{R}$	$f : \mathbb{R}^2 \rightarrow \mathbb{R}^2$	-

## 16.4 Semantics

### 16.4.1 State Variables

### 16.4.2 Access Routine Semantics

*# Calculate the gradient of a 2D function  $f$*

$\text{grad}(f)$ :

- output:  $\nabla f(x, y)$  such that

$$\forall (x, y) \in \mathbb{R}^2, \nabla f(x, y) = \begin{bmatrix} \frac{\partial f}{\partial x}(x, y) \\ \frac{\partial f}{\partial y}(x, y) \end{bmatrix}$$

- exception:

## 17 MIS of Least Square Fit Method Module (M 13)

*# 2D linear least square method to fit a function  $f$*

### 17.1 Module

LSFMCalc

### 17.2 Uses

Data Structure

### 17.3 Syntax

#### 17.3.1 Exported Access Programs

Name	In	Out	Exceptions
lsfm	$f : \mathbb{R}^2 \rightarrow \mathbb{R}^2, U$	$f : \mathbb{R}^2 \rightarrow \mathbb{R}^2$	-

## 17.4 Semantics

### 17.4.1 State Variables

### 17.4.2 Access Routine Semantics

*# Calculate the 2D fit of a function  $f$  using the linear least square method on a domain  $U = ([x_0, x_1]; [y_0, y_1]) \in \mathbb{R}^2$*

$\text{lsfm}(f, U)$ :

- output:  $fit(x, y) = ax + by$  such that

$$\begin{aligned} \forall (x, y) \in U, E(a, b) &= \int_{x_0}^{x_1} \int_{y_0}^{y_1} [f(x, y) - fit(x, y)]^2 dx dy \text{ is minimized} \\ \Rightarrow \frac{\partial E}{\partial a} = 0 \wedge \frac{\partial E}{\partial b} = 0 &\Rightarrow a = \frac{\int_{x_0}^{x_1} \int_{y_0}^{y_1} x f(x, y) dx dy}{\int_{x_0}^{x_1} \int_{y_0}^{y_1} x^2 dx dy} \wedge b = \frac{\int_{x_0}^{x_1} \int_{y_0}^{y_1} y f(x, y) dx dy}{\int_{x_0}^{x_1} \int_{y_0}^{y_1} y^2 dx dy} \end{aligned}$$

- exception:

## 18 MIS of Phase Operation Module (M 14)

### 18.1 Module

PhaseCalc

### 18.2 Uses

Data Structure

### 18.3 Syntax

#### 18.3.1 Exported Access Programs

Name	In	Out	Exceptions
unwrap	$f : \mathbb{R}^2 \rightarrow ] - \pi, \pi]$	$f : \mathbb{R}^2 \rightarrow \mathbb{R}$	-
wrap	$f : \mathbb{R}^2 \rightarrow \mathbb{R}$	$f : \mathbb{R}^2 \rightarrow ] - \pi, \pi]$	-
arg	$z \in \mathbb{C}$	$\phi \in ] - \pi, \pi]$	

### 18.4 Semantics

#### 18.4.1 State Variables

#### 18.4.2 Access Routine Semantics

$wrap(f)$ :

- output:  $g$  such that

$$\forall (x, y) \in \mathbb{R}^2, \exists k \in \mathbb{Z} | g(x, y) = f(x, y) + 2k\pi \wedge g(x, y) \in ] - \pi, \pi]$$

- exception:

$unwrap(f)$ :

- output:  $g$  such that

$$\begin{aligned} & \forall (x, y) \in \mathbb{R}^2, \exists k \in \mathbb{Z} | g(x, y) = f(x, y) + 2k\pi \wedge g \text{ is continuous} \\ \Rightarrow & \forall (x, y) \in \mathbb{R}^2, \exists k \in \mathbb{Z} | \lim_{(x,y) \rightarrow (x_0,y_0)} g(x, y) = g(x_0, y_0) = f(x_0, y_0) + 2k\pi \end{aligned}$$

- exception:

$\arg(z)$ :

- output:  $\phi$  such that

$$\phi = \arg(z) \text{ with } z = e^{i\phi}$$

- exception:

## 19 MIS of Data Structure Module (M 15)

### 19.1 Module

DataStruct

### 19.2 Uses

None

### 19.3 Syntax

#### 19.3.1 Exported Access Programs

Name	In	Out	Exceptions
store	string $\times$ object	-	-
read	string	object	-

### 19.4 Semantics

#### 19.4.1 State Variables

*# Structure of the object carrying the data information*

data : object

- $\text{data}(\text{ISMHexp}) = I_{SMH_{\text{exp}}}$
- $\text{data}(\text{pISMHexp}) = p$
- $\text{data}(\text{ICref}) = I_{C_{\text{ref}}}$

- $\text{data}(\text{pICref}) = p_{\text{ref}}$
- $\text{data}(\text{FTISMHexp}) = \tilde{I}_{SMH_{\text{exp}}}$
- $\text{data}(\text{FTISMHsim}) = \tilde{I}_{SMH_{\text{sim}}}$
- for each  $j$   $\text{data}(\text{T}j)$  : object
  - $\text{data}(\text{T}j)(\text{gMuns}) = \vec{g}_j^{M_{\text{exp}}}_{\text{uns}}$
  - $\text{data}(\text{T}j)(\text{deltagM}) = \Delta \vec{g}_j^{M_{\text{exp}}}$
  - $\text{data}(\text{T}j)(\text{PhasegM}) = P_{\Delta \vec{g}_j^{M_{\text{exp}}}}$
  - $\text{data}(\text{T}j)(\text{shift}) = (n_j, m_j)$
  - $\text{data}(\text{T}j)(\text{gCuns}) = \vec{g}_j^{C_{\text{exp}}}_{\text{uns}}$

### 19.4.2 Access Routine Semantics

$\text{store}(a, b)$ :

- transition:  $\text{data}(a) = b$

$\text{load}(a)$ :

- output:  $\text{data}(a)$

## 20 MIS of Generic GUI/Plot Module (M 16)

### 20.1 Module

GUIGene

### 20.2 Uses

Hardware-Hiding Data Structure

### 20.3 Syntax

#### 20.3.1 Exported Access Programs

Name	In	Out	Exceptions
plot	GUI objects	-	-
fig	$\text{string} \times \text{GUI objects}$	GUI object	-
button	$k \in \mathbb{N}$ , $\text{string}^k$	GUI object	-
entry_field	string	GUI object	-
circle	-	GUI object	-
rectangle	-	GUI object	-
read_user_GUI	GUI object	object	-

## 20.4 Semantics

### 20.4.1 State Variables

### 20.4.2 Access Routine Semantics

plot():

- transition:
- output: Display on the Hardware all the GUI objects

fig('label', optional GUI objects):

- transition:
- output: Create a window GUI object with the optional GUI objects

button(*number*, 'labels'):

- transition:
- output: Create *number* buttons GUI objects with their respective 'labels'

entry\_field(*b*):

- transition:
- output: Create a entry field GUI object to collect the input *b* from the user

circle(*C*(user\_param)):

- transition:
- output: Create a circle *C* GUI object drawn by the user

rectangle(*R*(user\_param)):

- transition:
- output: Create a rectangle *R* GUI object drawn by the user



`read_user_GUI( $A$ ):`

- output:  $B$  such that  $B$  includes the id of the GUI and the type of the GUI

## 21 Appendix

All variables

$$\begin{aligned}
&P_{\Delta \vec{g}_j^{M_{\text{exp}}}(\vec{r})}, \vec{g}_j^{M_{\text{exp}}}, \Delta \vec{g}_j^{M_{\text{exp}}}(\vec{r}) \\
&\Delta \vec{g}_j^{M_{\text{exp}}}(\vec{r}), U, \vec{g}_j^{M_{\text{exp}}} \\
&\vec{g}_j^{M_{\text{exp}}}, \Delta \vec{g}_j^{M_{\text{exp}}}(\vec{r}) \\
&\vec{g}_j^{M_{\text{exp}}}, \Delta \vec{g}_j^{M_{\text{exp}}}(\vec{r}), \overrightarrow{q_{n_j, m_j}}, p \\
&\Delta \vec{g}_j^{C_{\text{exp}}}(\vec{r}), \vec{g}_j^{C_{\text{exp}}}
\end{aligned}$$