Project Title: System Verification and Validation Plan for Sun Catcher

Sharon(Yu-Shiuan) Wu

November 4, 2019

1 Revision History

Date	Version	Notes
2019/11/04 1	1.0	First version of System VnV
Date 2	1.1	-

Contents

List of Tables

List of Figures

2 Symbols, Abbreviations and Acronyms

symbol	**description**
T	Test
SC	Sun Catcher
Φ_P	the latituade of the solar panel
δ_{date}	the declination of the vertical noon sun in the day
I_S	the intensity of the sun measured by the satellites
P_{A_h}	the height of the solar panel
P_{A_w}	the width of the solar panel
$date$	the date duration between the calculation starting date and the ending date
$year_{\text{Start}}$	the year of the calcuation's starting date
$month_{\text{Start}}$	the month of the calcuation's starting date
day_{Start}	the day of the calcuation's starting date
$year_{\text{End}}$	the year of the calcuation's ending date
$month_{\text{End}}$	the month of the calcuation's ending date
day_{End}	the day of the calcuation's ending date

This document provides an outline of the system verification and validation for the Sun Catcher. The general introduction section provides readers a summary of the functions in Sun Catcherand the related documents as the resources for testing. The plan section provides readers the plan for verifying and validatingSC's software requirements specification(SRS) and introduces the method, tools, and external data to implement the testing. The system test description providesthe readers with the test cases related to functional and nonfunctional requirements in Sun Catcher. The test builds for uncovering the errors and boosting the confidence of the software while ensuring an acceptable performanceis provided.

3 General Information

3.1 Summary

The subsection belows are the test cases of Sun Catcher. Sun Catcher is the software that calculate the optimum tilt angle of the date duration that decidedby users. Then it takes the calculated angle to estimate the optimal solar energy output.

3.2 Objectives

The goal of the test is to build confidence in the software correctness and strengthen the robustness of the software. The functional and nonfunctional requirements of Sun Catcher are tested by this Verification and Validation Plan. The equations that related to the functional requirements are found in the SRS [1].

The functional requirements are:

R1: Sun Catcher should able to read the user's input value from the system's interface.

R2: Sun Catcher should able to output the result in the corrected format.

R3: Sun Catcher should able to calculate the date duration between the calculation starting date and the ending date.

R4: Sun Catcher should able to calculate the optimum tilt angle of the date duration.

R5: Sun Catcher should able to calculate the produced solar energy satisfied the date duration.

R6: Sun Catcher should able to show the difference in produced solar energy satisfied the different date durations.

R7: Verify the output.

R8: The system should output the calculated results of the outcomes in a manner consistent with decision making.

The nonfunctional requirements are:

NFR1. Correct: The output should give correct results.

NFR2. Verifiable: The code is tested with complete verification and validationplan

NFR3. Understandable: The code is understandable for readers.

NFR4. Reusable: The code is modularized

NFR5. Maintainable: The traceability between requirements, assumptions, theoretical models, general definitions, data definitions, instance models, likely changes, unlikely changes, and modules is completely recorded in traceability matrices in the SRS and module guide

NFR6. Portable: The code is able to be run in different environments

3.3 Relevant Documentation

The SRS of the Sun Catchercan be found in [Wu [1]]

The external documents for verifying the equations used in Sun Catcher can be found in [Landau [2]] and [MarkandVijaysinh[3]]

4 Plan

4.1 Verification and Validation Team

The test team includes the following members:
Main reviewer: Sharon Wu

4.2 SRS Verification Plan

- Get feedback from the reviewers: Sasha Soraine, Zhi Zhang, and Doctor Smith, after SRS is completed and put to the GitHub.
- Check the document by using SRS-Checklist and Writing-Checklist before publishing to GitHub.

4.3 Design Verification Plan

- The design should be verified by complete and success the test cases in the system VnV plan under the section 5.
- The design should satisfy all the functional and nonfunctional requirementthat stated in the SRS document [1].

4.4 Implementation Verification Plan

The following tools will be used to facilitate testing:

Rubber Duck Debugging: Performed by author, Yu-Shiuan Wu. The author should verbally explain the code line by line.

Haskell Program Coverage: Dynamic Testing Tool, a tool-kit to record and display the code coverage of a Haskell Program. It aims to reinforce the correctness of the software and to eliminate the infeasibility problems.(Gill and Runciman[4])

QuickCheck: Automatic testing tool for Haskell programs, a library for random testing of program properties. It aims to boost the robustness of the software.(Claessen[5])

4.5 Software Validation Plan

Sun Catchershould be valid by satisfied all the functional requirement in SRS plan.
Based on the physical concept of Sun Catcher, the author, Yu-Shiauan Wu, should record the actual solar energy by using the output from Sun Catcher. Then verify whether the calculated tilt angle can increase the energy gaining.

5 System Test Description

5.1 Tests for Functional Requirements

The subsection below is designed to cover the functional requirements of Sun Catcher, which also describes in section 3.2.
The test is divided into four subsections, which are input reading, input bounds, output calculation, and output verification. Input reading testing is designed for testing the ability to receive information from the software interface. Input bounds testing and output calculation testing are designed for testing the robustness of the software. Output verification testing is designed for the correctness of the implemented equation.

5.1.1 Input Reading

This test covers the requirements, R1, in section 3.2. Based on the SRS document[1], Sun Catcherhas to identity users' inputs and then assign the values to designated equations or modules.

Identity users' input

1. InputReading-id1

   Control: Manual. Input the input value from the keyboard.

   Initial State: No input value

   Input: Input the value of requirements of Sun Catcher. It required to input the value of latitude, the area of the solar panel, the day started to estimate angle and day when the estimation end.

The given inputs are:
Φ_P: 43.250943 P_{A_h}:1455 P_{A_w}:665 $year_{\text{Start}}$:(2019) $month_{\text{Start}}$:(01) day_{Start}:(01) $year_{\text{End}}$:(2019) $month_{\text{End}}$:(12) day_{End}:(31)

Output: The expected result wil for the given inputs is:
Φ_P: 43.250943 P_{A_h}:1455 P_{A_w}:665 $year_{\text{Start}}$:(2019) $month_{\text{Start}}$:(01) day_{Start}:(01) $year_{\text{End}}$:(2019) $month_{\text{End}}$:(12) day_{End}:(31)

Test Case Derivation: The output is justified if the output value is equal to the corresponding input value.

How test will be performed:

- Input the value from the keyboard following the instruction of the software interface.
- Verified the output showing on the screen by the test case derivation instruction.

5.1.2 Input Bounds

This test covers the requirements, R1, in section 3.2. Based on the SRS document[1], the input data constraints can be found in the table, Specification Parameter Values, under the section, Input Data Constraints.

The Robustness of Input Bounds

1. InputBounds-id2

   Control: Manual.

   Input the extreme ends of the value of latitude. Initial State: No input value Input: Based on SRS[1] of the Sun Catcher[1], the boundaried of latitude is $-90 \leq \Phi_P \leq 90$.
   Therefore, the given inputs is:
   input (1) Φ_P: 90 input (2) Φ_P: 90 input (3) Φ_P: 91 input(4) Φ_P: -91

   Output: The expected result wil for the given inputs is:
   output (1) Φ_P: 90 output (2) Φ_P: 90 output (3) Φ_P: Latitude is not

greater than 90 output (4) Φ_P: Latitude is not less than -90

Test Case Derivation: The output is justified if the output value is equal to the corresponding input value. When the input value is over the boundary of latitude, the system activates the error handler.

How test will be performed:

- Input the value from the keyboard following the instruction of the software interface.
- Verified the output showing on the screen by the test case derivation instruction.

2. InputBounds-id3
   Control: Manual. Input the extreme ends of the value of calendar. Initial State: $year_{\text{Start}}$:(2019) $month_{\text{Start}}$:(02) $year_{\text{End}}$:(2020) $month_{\text{End}}$:(02)

   Input: Based on calendar, the given inputs is:
   Input(1) day_{Start}:(28) day_{End}:(28)
   Input(2) day_{Start}:(29) day_{End}:(29)

   Output:The expected result wil for the given inputs is:
   Output(1) day_{Start}:(28) day_{End}:(28)
   Output(2) 2020.02.29 does not exist

   Test Case Derivation: The output is justified if the output value is equal to the corresponding input value. When the input integer is over the boundary of the calendar, the system activates the error handler.

   How test will be performed:

   - Input the value from the keyboard following the instruction of the software interface.
   - Verified the output showing on the screen by the test case derivation instruction.

3. InputBounds-id4
   Control: Manual. Input the extreme ends of the value of the Gregorian calendar. Initial State: No any given value. Input: Based on the calendar, the given inputs is:
   Input (1) $year_{\text{Start}}$:(0) $month_{\text{Start}}$:(0) day_{Start}:(0)
   Input (2) $year_{\text{Start}}$:(-1) $month_{\text{Start}}$:(-1) day_{Start}:(-1)

   Output: The expected result wil for the given inputs is:
   Output (1) 0,0,0 doesnot exist.
   Output (2) -1,-1,-1 does not exist.

   Test Case Derivation: When any input value less than equal than 0, the system activates the error handler.

   How test will be performed:

   - Input the value from the keyboard following the instruction of the software interface.
   - Verified the output showing on the screen by the test case derivation instruction.

5.1.3 Output Calculation

This test covers the requirements, R3 to R6, in section 3.2. This test relates to the previous test input reading testing. After the system reads the inputs from the software interface, the system starts calculating the outputs.

The Robustness of the Calculation

1. CalculateOutput-id5

   Control: Automatic. Input the random cases that satisfied the values' property. Initial State: No input value

   Input: Input the value of requirements of Sun Catcherthat drive the calculation of the date duration. To calculate the date duration, it needs $year_{\text{Start}}$ $month_{\text{Start}}$ day_{Start} $year_{\text{End}}$ $month_{\text{End}}$ day_{End}

The given inputs are:
Input (1..50) $year_{\text{Start}}$: A random picked integer
$month_{\text{Start}}$: A random picked integer
day_{Start}: A random picked integer
$year_{\text{End}}$: A random picked integer
$month_{\text{End}}$: A random picked integer
day_{End}: A random picked integer

Output: The expected result wil for the given inputs is:
Output (1..50) : Show“OK, passed 49 tests ” on the screen

Test Case Derivation: Based on the description in QuickCheck[5], the output is justified if the output shows `‘‘OK, passed input-numbers of tests ’’`

How test will be performed:

- Implement the test cases with QuickCheck[5], describes in section 4.3.
- Verified the output showing on the screen by the test case derivation instruction.

2. CalculateOutput-id6
   Control: Automatic. Input the random cases that satisfied the values' property.

   Initial State: No input value

   Input: Input the value of requirements of Sun Catcherthat drive the calculation of the zenith angle. To calculate the zenith angle, it needs Φ_P and δ_{date}

   The given inputs are:
   Input (1..50) Φ_P: A random picked rational number
   δ_{date}: A random picked integer

   Output: The expected result wil for the given inputs is:
   Output (1..50) : Show`‘‘OK, passed 49 tests ’’` on the screen

Test Case Derivation: Based on the description in QuickCheck[5], the output is justified if the output shows `‘‘OK, passed input-numbers of tests ’’`

How test will be performed:

- Implement the test cases with QuickCheck[5], describes in section 4.3.
- Verified the output showing on the screen by the test case derivation instruction.

3. CalculateOutput-id7
   Control: Automatic. Input the random cases that satisfied the values' property.

   Initial State: Based on the assumption in SRS[1], $I_S = 1.35$

   Input: Input the value of requirements of Sun Catcherthat drive the calculation of the Solar intensity. To calculate the solar intensity, it needs $\theta_{S_{day}}$

   The given inputs are:
   Input (1..50) $\theta_{S_{day}}$: A random picked rational number

   Output: The expected result wil for the given inputs is:
   Output (1..50) : Show`‘‘OK, passed 49 tests ’’` on the screen

   Test Case Derivation: Based on the description in QuickCheck[5], the output is justified if the output shows `‘‘OK, passed input-numbers of tests ’’`

   How test will be performed:

   - Implement the test cases with QuickCheck[5], describes in section 4.3.
   - Verified the output showing on the screen by the test case derivation instruction.

5.1.4 Output Verification

This test covers the requirements, R2, R7 and R8, in section 3.2. This test uses external data3.3 to verify the output. Based on the SRS document[1], the output data constraints can be found in the table, Output Variables, under the section, Properties of a Correct Solution.

The Correctness of the Calculation

1. VerifyOutput-id8
   Control: Automatic. The test cases contain cases that $|year_{\text{Start}} - year_{\text{End}}| = 0$ and $|year_{\text{Start}} - year_{\text{End}}| \geq 0$.

   Initial State:No input value

   Input: Input the value of requirements of Sun Catcherthat drive the calculation of the date duration. To calculate the date duration, it needs $year_{\text{Start}}$ $month_{\text{Start}}$ day_{Start} $year_{\text{End}}$ $month_{\text{End}}$ day_{End}

   The given inputs are:
   Input (1) $year_{\text{Start}}$: 2017 $month_{\text{Start}}$: 01 day_{Start}: 01 $year_{\text{End}}$: 2017 $month_{\text{End}}$: 12 day_{End}: 31

   Input (2) $year_{\text{Start}}$: 2018 $month_{\text{Start}}$: 06 day_{Start}: 20 $year_{\text{End}}$: 2019 $month_{\text{End}}$: 07 day_{End}: 26

   Output: The expected result wil for the given inputs is the date duration:
   Output (1) : 365 Output (2) : 402

   Test Case Derivation: Based on the calendar, absolute error = 0 where absolute error = | expected output − actual output| How test will be performed:

   - Input the input values from a file, VerifyOutputId1.txt.
   - Verified the output by the test case derivation instruction.
   - If all the absolute error of the test cases is equal to 0, then the test success, otherwise the test fails.

2. VerifyOutput-id9

   Control: Automatic. The test cases contain cases that
   $0 \leq \Phi_P,\ \delta_{\text{date}} \leq 90$,
   $-90 \leq \Phi_P,\ \delta_{\text{date}} < 0$,

$-90 \leq \delta_{\text{date}} < 0 \ \wedge \ 0 \leq \Phi_P \leq 90$,
$-90 \leq \Phi_P < 0 \ \wedge \ 0 \leq \delta_{\text{date}} \leq 90$

Initial State: No input value

Input: Input the value of requirements of Sun Catcherthat drive the calculation of the zenith angle. To calculate the zenith angle, it needs Φ_P and δ_{date}

The given inputs are:
Input (1) Φ_P: 43.250943 δ_{date}: 20
Input (2) Φ_P: -43.250943 δ_{date}: -20
Input (3) Φ_P: -43.250943 δ_{date}: 20
Input (4) Φ_P: 43.250943 δ_{date}: -20

Output: The expected result wil for the given inputs is the zenith angle:
Output (1) 23.250943
Output (2) 23.250943
Output (3) 63.250943
Output (4) 63.250943

Test Case Derivation: Based on the equation described in SRS[1], we get the expected result. Therefore, absolute error = 0 where absolute error = $|$ expected output $-$ actual output$|$,

How test will be performed:

- Input the input values from a file, VerifyOutputId2.txt.
- Verified the output by the test case derivation instruction.
- If all the absolute error of the test cases is equal to 0, then the test success, otherwise the test fails.

3. VerifyOutput-id10

This test case is driven by the previous test case, VerifyOutput-id2. Therefore,if the previous test case success, VerifyOutput-id3is implemented.
Moreover, this test case used the external data from [Jacobsonand

Jadhav[3]] as the expected output and the expected input latitude.

Control: Automatic. The test cases contain cases that Φ_P = expected input latitude and $\Phi_P \neq$ expected input latitude.

Initial State: Based on the assumption in SRS[1], I_S:1.35, and based on the assumption in [Jacobson and Jadhav[3]], $year_{\text{Start}}$: 2018 $month_{\text{Start}}$: 01 day_{Start}: 01; $year_{\text{End}}$: 2018 $month_{\text{End}}$: 12 day_{End}: 31

Input: Input the value of requirements of Sun Catcherthat driven the calculation of the optimal tilt angle. To calculate the solar intensity, it needs $\theta_{S_{day}}$, which is diven by the input latitude.

The given inputs are:
Input (1)Φ_P: 64.13
Input(2)Φ_P:63.13

Output: The expected result wil for the given inputs is the optimal tilt angle:
Output (1) : 43
Output (2) : 43

Test Case Derivation: Based on the equation described in SRS[1], we get the **actual result**. Then we calculate the relative error using the data in the [JacobsonandJadhav[3]] as our **expected result**. Therefore, relative error ≈ 0 where relative error $= |1 - \frac{\text{actual result}}{\text{expected result}}|$

How test will be performed:

- Build a linear graph using the expected input latitude as the x-axis and expected output as the y-axis.
- Input the input values from a file, VerifyOutputId3.txt.
- Calculate the **actual result** by the equation descibes in in SRS[1]
- Place the point($P_{\text{actual input}}$)(x-axis: input latitude, y-axis: actual result) in the linear graph
- Find the point($P_{\text{upper bound}}$),the lowest upper bound of $P_{\text{actual input}}$ and point($P_{\text{lower bound}}$), the greatest upper bound of $P_{\text{actual input}}$

- Calculate the area between $P_{\text{upper bound}}$ and $P_{\text{actual input}}$; and $P_{\text{lower bound}}$ and $P_{\text{actual input}}$ using the equation,
$Area = \frac{|(x_{\text{input latitude}} - x_{\text{expected latitude}})| \times |(y_{\text{actual result}} - y_{\text{expected result}})|}{2}$
- If $Area_{\text{actual input - } \textbf{upper bound}} < Area_{\text{actual input - } \textbf{lower bound}}$, then **expected result** = the y-axis of $P_{\text{upper bound}}$, otherwise **expected result** = the y-axis of $P_{\text{lower bound}}$
- Verified the output by the test case derivation instruction.
- If all the relative error of the test cases is approximately 0, then the test success, otherwise the test fails.

4. VerifyOutput-id11

This test case is driven by the previous test case, VerifyOutput-id1, VerifyOutput-id2 and VerifyOutput-id3. Therefore, if the previous test case success, VerifyOutput-id4 is implemented.

Moreover, this test case used the external data from [Landau[2]] as the expected output, expected input latitude

Control: Automatic. The test cases contain cases that Φ_P = expected input latitude, $\Phi_P \neq$ expected input latitude.

Initial State: Basedon the assumption in SRS[1], I_S: 1.35, and based on the assumption in [Landau[2]], the date duration of the winter in northern hemisphere is from
$year_{\text{Start}}$: 2018 $month_{\text{Start}}$: 10 day_{Start}: 05
to $year_{\text{End}}$: 2019 $month_{\text{End}}$: 03 day_{End}: 05

Input: Input the value of requirements of Sun Catcherthat driven the calculation of the Solar intensity. To calculate the solar intensity, it needs $\theta_{S_{day}}$, which is diven by the input latitude.

The given inputs are:
Input (1)Φ_P: 30
Input (2)Φ_P: 31

Output: The expected result wil for the given inputs is average the solar intensity during winter:
Output (1) : 5.6
Output (2) : 5.6

Test Case Derivation: Based on the equation described in SRS[1], we get the expected result. Therefore, relative error ≈ 0 where relative error $= |1 - \frac{\text{actual output}}{\text{expected output}}|$

How test will be performed:

- Input the input values from a file, VerifyOutputId4.txt.
- Calculate the daily solar intensity during winter
- Calculate the average solar intensity during winter, using the equation,
  the average solar intensity $= \frac{\text{the sum of the daily solar intensity}}{\text{date duration of winter}}$
- Output the average solar intensity as the actual output
- Verified the output by the test case derivation instruction.
- If all the relative error of the test cases is approximately 0, then the test success, otherwise the test fails.

5.2 Tests for Nonfunctional Requirements

5.2.1 Correctness

The correctness of the System

1. correctness-id1

   Type: Dynamic analysis The outputs of the system test under section 5.1.4. To get a reliable output, Sun Catcheris expected to pass all the test cases under the section 5.1.4.

   How test will be performed: Active the test cases under section 5.1.4, when the code is modified.

2. correctness-id2

Type: Dynamic analysis - Code Coverage

Use the code coverage tool, Haskell Program Coverage(HPC), to test the code coverage. The description of HPC can be found in section 4.3.

Input/Condition: The main code of Sun Catcher

Output: The percentage of the coverage

How test will be performed:

- Use the compiler, ghc-6.8.1 or later version, to active Haskell Program Coverage.
- Enable hpc with the command line, -fhpc.
- The test case success, if the output gets the 100otherwise the test case fails.

5.2.2 Portability

The portability of the system

1. portability-id3

   Type: Manual

   Initial State: -

   Input/Condition: Implement Sun Catcherin diverse environments.

   Output/Result: If the Sun Catcherworks functionally.

   How test will be performed:

   - Implement Sun Catcher on the virtual machines with the system enviroment of Windows 10, MacOs.
   - Run every function of Sun Catcher.
   - If function works, then success, otherwise fail.

5.3 Traceability Between Test Cases and Requirements

	R1	R2	R3	R4	R5	R6	R7	R8
id1	X							
id2	X							
id3	X							
id4	X							
id5			X					
id6				X				
id7					X			
id8							X	X
id9			X				X	X
id10		X		X		X	X	X
id11				X			X	X

Table 1: Traceability Between Functional Requirements Test Cases and Requirements

	NFR1	NFR2	NFR3	NFR4	NFR5	NFR6
id1	X					
id2	X					
id3						X

Table 2: Traceability Between NonFunctional Requirements Test Cases and Requirements

References

[1] Yu-Shiuan Wu. Software requirements speci
cation for sun catcher, 2019. URL https://github.com/sharyuwu/optimum-tilt-of-solar-panels/blob/master/docs/SRS/SRS.pdf.

[2] Charles R. Landau. Optimum tilt of solar panels, 2001. URL https://www.solarpaneltilt.com/#other.

[3] Mark Z. Jacobson and Vijaysinh Jadhav. World estimates of pv optimal tilt angles and ratios of sunlight incident upon tilted and tracked pv panels relative to horizontal panels. Technical Report CAS-17-01-SS, Department of Civil and Environmental Engineering, Stanford University, Stanford,USA, 2018. URL https://web.stanford.edu/group/efmh/jacobson/Articles/I/TiltAngles.pdf.

[4] Andy Gill and Colin Runciman. Haskell program coverage, 2000. URL https://wiki.haskell.org/Haskell_program_coverage.

[5] Koen Claessen. Quickcheck: Automatic testing of haskell programs, 2000. URL http://hackage.haskell.org/package/QuickCheck.

6 Appendix

6.1 Symbolic Parameters

Symbol	**Description**	**Value**
I_S	Solar insensity	1.35

6.2 Usability Survey Questions?

Symbol	**Answer**
Do you think this software helps?	Yes/ No
Do you think this software is easy to use?	Yes/ No
Do you think this software is easy to use for elders?	Yes/ No
Do you think this software is easy to use for children(under 12)?	Yes/ No
Do you think this software works flawlessly?	Yes/ No
How many time you open this software in a week?	_________times
How many stars would you like to give to this software?(1 - 10 starts)	_________starts
Do you like to recommend this software to others?	Yes/ No

Table 3: Survey for home users

Symbol	**Answer**
The steps take to get the optimum tilt angle	_________steps
How long it need to get the optimum tilt angle	_________times
The steps take to get the expected solar energy gaining	_________steps
How long it need to get the expected solar energy gaining	_________times
The power consume after using the software for an hour	_________%
The memory consume after install the software	_________%
How many error alerts you take at the fist time using the software?	_________times
How many error alerts you take at the second time using the software?	_________times

Table 4: Survey for researching group