ARTIFICE Construct

Public Release

Terminology has been deliberately obfuscated in this document. Certain keywords have been substituted. Full documentation will (most likely) not be made publicly available

1 What is a Construct?

A construct is a network of people and locations. Each person is given several characteristics, including but not limited to height, weight, date of birth, subgroup, residency, and profession.

1.1 Classifying a Person

Every person belongs to a class. There are four classes - **Main, Secondary, Support, Outsider**. We are primarily concerned with the main group, as they have the most groupings and characteristics.

1.1.1 Main Class Sub-Groupings

Within the Main class, each person belongs to one of 4 groups: A, X, S, or H; the H group stands for Hybrid, where they share characteristics of 2 or more of the A, X, or S groups. Each person also has a "Level", which is a letter from A to H or the number 0, determined by the 14-byte string.

Each Main class person also has 14 or 21 numbers associated with it, which we will explore in greater detail later.

1.2 Terminology Used

- 14-string / 21-string: Describes the specific characteristics of a Main class person.
- Level: 0 or A - H.
- Group: A, X, S, or H.
- Mode: The current state of a main class person. Can be one of: **Zone, Chair, Flat, Struct**.

2 Class and Levels

As mentioned above, each main class person has a "level" in [0A-H] and a class out of A, X, S, and H. The following rules exist for creating people:

1. Let X-Y denote a person with level X and class Y. For example, G-S would denote a person with level G, class S.
2. Class A people may only have levels 0 and B.
3. Class S people cannot have level 0.

2.1 A Class

A-class people can be described by a 4 character string, where each character is one of A, B, C, or 0. We call this person **symmetric** if str[0] == str[1] and str[2] == str[3]. For example, "BB00" would be symmetric, while "AA0C" would not be.

2.1.1 Symmetric conversion to 14-string

The first character determines the first 3 characters of the string. We have:

- 0: First 3 characters are 000
- C: First 3 characters are 200
- B: First 3 characters are 400
- A: First 3 characters are 440

The second character determines characters 5 through 8. We have:

- 0: 0000
- C: 4000
- B: 4400
- A: 4442

The rest of the characters are to be filled with 0. For example, "BBAA" would be converted to 4000-44420-000-00.

2.1.2 Asymmetric conversion to 14-string

For an asymmetric A-classer, the 13th character will be 2. Characters 1-3 and 5-8 would be determined by str[0] and str[2]. Then following the 14th character, 7 more characters would be appended, determined by str[1] and str[3].

For example, "BABC" would be converted to 4000-44000-000-20-4404000.

In the event that one side is completely 0 (example "B0B0"), we make the 13th character 1 and stick to the 14-string.

2.2 S Class

The S-Class is a bit more rigid. There are few set archetypes, and a little bit of variation within the archetypes. The available archetypes are:

- T1 through T12 (for example, T8 is valid, but T13 is not).
- C1 through C7
- 1A, 1B, 1C, 2A, or 2B
- CX11, CX12, or CX5
- C1V, C1VCX5, or C1-H

We also assign each person a "seed" number from 0 to 9. Let X.y denote level X with seed y; for example, T8.2 would denote T8 with seed 2.

And the corresponding 14-strings are:

- T8-T12 and T7.0-4: 4442-00000-000-00
- T7.5-9, T6: 4443-00000-000-00
- T2-T5: 4444-00000-000-00
- T1, C7.0-1: 4444-10000-000-00
- C7.2-6: 4444-20000-000-00
- C7.7-9: 4444-20100-000-00
- C6:
  - C6.0: 4444-21101-000-00
  - C6.1: 4444-31101-000-00
  - C6.2: 4444-41101-000-00
  - C6.3: 4444-21202-000-00
  - C6.4-5: 4444-31202-000-00
  - C6.6: 4444-41202-000-00
  - C6.7: 4444-21302-000-00
  - C6.8: 4444-31302-000-00
  - C6.9: 4444-41302-000-00
- C5:
  - C5.0-1: 4444-41312-000-00
  - C5.2-4: 4444-42312-000-00
  - C5.5-7: 4444-41313-000-00

  - C5.8-9: 4444-42313-000-00

- C4, C3.0-3: 4444-44444-000-00
- C3.4-9: 4444-44444-100-00
- C2, C1.0-5: 4444-44444-200-00
- C1.6-8: 4444-44444-300-00
- C1.9: 4444-44444-301-00
- 2B: 4444-12333-000-00
- 2A.0-4: 4444-11344-100-00
- 2A.5-9: 4444-12344-100-00
- 1C.0-4: 4444-44444-012-00
- 1C.5-9: 4444-44444-112-00
- 1B.0: 4444-44444-212-00
- 1B.1-4: 4444-44444-312-00
- 1B.5-9: 4444-44444-412-00
- 1A.0-2: 4444-44444-413-00
- 1A.3-9: 4444-44444-414-00
- CX11.0-7: 4444-44444-400-00
- CX11.8-9: 4444-44444-401-00
- CX12.0-5: 4444-44444-302-00
- CX12.6-9: 4444-44444-402-00
- CX5.0-4: 4444-44444-303-00
- CX5.5-9: 4444-44444-403-00
- C1V.0-4: 4444-44444-210-00
- C1V.5-8: 4444-44444-310-00
- C1V.9: 4444-44444-410-00
- C1VCX5.0-1: 4444-44444-313-00
- C1VCX5.2-9: 4444-44444-413-00
- C1H: 4444-44444-404-00

2.3 X Classes

We can describe an X class person with 7 components. Call these components **G, M, A, H, F, V, X**. These components take on the following values:

- G: 1, 2a, 2b, 3a, 3b, 4a, 4b, 5a, 5b, 5c
- M: 1, 2a, 2b, 3a, 3b, 3c, 4a, 4b, 5
- A: 1, 2, 3, 4a, 4b, 4c, 5a, 5b, 5c
- H: 1, 2, 3, 4, 5
- F: 1, 2, 3-, 3, 4, 5
- V: 0, 1
- X: 0, 1, 2

The values are assigned according to the following rules:

- G to M rules
  - If G is less than or equal to 4a, then MA must be 11.
  - If G is 4b, M must be at most 4b.
  - If G is 5a, M must be at most 3a.
  - If G is 5b, M must be from 3b to 4b.
  - If G is 5c, M must be at least 4a.
- M to A rules
  - If M is 1, A must be 1 or 2.
  - If M is 2a or 2b, A must be 1, 2, or 3.
  - If M is in 3a-3c, A must be in 3-4c.
  - If M is 4a, A must be 4a or 4b.
  - If M is 4b, A must be in 4a-5c.
  - If M is 5, A must be 5c.
- HFVX must be 1100 unless A is 5c.

We define a X-class person as **lower** if GMA is not 5c55c or if their HFV is equal to 110. We define them as **upper** otherwise.

Like the S-class, we assign a seed with values from 0 - 9. We assign values to the 14-string as follows, for **Lower** X-Classes:

- G Values: First 4 numbers
  - 1: 1000

  - 2a: 2000
  - 2b: 3200
  - 3a: 4310
  - 3b: 4420
  - 4a: 4440
  - 4b: 4441
  - 5a: 4442
  - 5b: 4443
  - 5c: 4444
- M Values: Numbers 5 and 6
  - 1: 00
  - 2a.0-4: 10
  - 2a.5-9: 20
  - 2b.0-2: 01
  - 2b.3-6: 11
  - 2b.7-9: 21
  - 3a.0-4: 22
  - 3a.5-9: 23
  - 3b: 40
  - 3c: 42
  - 4a.0-4: 33
  - 4a.5-9: 34
  - 4b: 43
  - 5: 44
- A Values: Numbers 7,8,9
  - 1: 000
  - 2: 001
  - 3: 111
  - 4a: 122
  - 4b: 123
  - 4c: 223
  - 5a: 333
  - 5b: 334
  - 5c: 444

Now, we consider upper X-classes. They have the following archetypes:

- F = 1
  - Type 11: 2D, 2x1D
  - Type 21: 2D, 1D
- F = 2
  - Type 12: 2D, 2x1D
  - Type 22: 2D, 1D
  - Type 32: 1D + 1
  - Type 42: 1D
- F = 3- or 3
  - Type 33: 1D + 1
  - Type 43: 1D
  - Type 53: S
- F = 4
  - Type 44: 1D
  - Type 54: S
- F = 5
  - Type 45: Roll
  - Type 55: H

Each archetype has the following builds:

- F determines the 12th value. It corresponds to F - 1; for example F4 would have 12th value 3.
- V determines the 11th value. V = 0 corresponds to 11th = 0. V can only be 1 if F is 4 or 5.
- Then we describe the 1st through 10th values.
- Like the A classes, we introduce asymmetry on values 1-3 and 5-8.
- Type 11, 12:
  - 4444-44444-0
  - 4444-44444-1...4444244
  - 4444-44444-1...2444444
  - 4444-42444-2...4442244

- Type 21, 22:
  - 4444-44444-1
  - 4444-44444-2...4442244
  - 3444-44444-4...2442244
- Type 32, 33:
  - 4444-44444-2
  - 4444-44444-3...3444444
  - 4444-44444-3...4444344
  - 4444-43444-4...4444244
  - 4444-44444-4...2444344
  - 4444-44444-4...3444244
- Type 42, 43, 44, 45:
  - 4444-44444-3
  - 4444-44444-4...2444444
  - 4444-44444-4...4444244
  - 4444-43444-4...4444344
  - 3444-44444-4...3444444
- Type 53, 54, 55:
  - 4444-44444-4...3444444
  - 4444-44444-4...4444344
  - 4444-44444-4...4444444

Additionally, note that for upper X-group archetypes, they can be endowed with an 8th descriptor, one of A or C. They can also not have an 8th descriptor at all. Note that the C descriptor is valid only for the types 42, 53, 44, 45, 54, and 55.

2.4 Hybrid Groups

2.4.1 A + S Hybrids

We describe an A + S hybrid with both the A 4 character string and the S descriptor. There are some caveats, though. For the A 4-character string, only the last 2 characters matter; "AA0C" is functionally equivalent to "B00C". For the S descriptor, we want to pick something between T1 to T12 or C5 to C7.

Then we construct the 14/21-string as follows:

- First 4 characters determined by the S descriptor.
- Next 4 characters determined by the maximum of the S and the A descriptors.
- Last x characters should be all 0.

For example, suppose we had a hybrid with descriptors "00AB" and "T3". Then we would construct 21-string 4442-44420-000-20-4444400.

2.4.2 A + X Hybrids

Similarly to above, we use the maximum of the A and the S descriptors. We also stick to lower X-classes rather than upper X-classes, because adding an A descriptor to an upper X class will not fundamentally change the 14 or 21-string.

For example, if we combined a GMA 5b3a4b with "00B0" and seed 3, we would have 21-string 4443-44123-000-20-4442212.

2.4.3 S + X Hybrids / A + S + X Hybrids

These are rare and should probably be avoided. In the event that you do want to use them, however, the same behavior applies with combining maximums. Stick to combining T1-T12/C5-C7 and lower X groups.

2.5 Enhancements

All A groupers, and A + S hybrids (excluding ASX hybrids), are eligible for enhancements. These enhancements change the 14 or 21-byte string, but are only active when the mode is **Chair**.

- First 2 characters:
  - 0: 000 → 000 (unchanged)
  - C: 200 → 000
  - B: 400 → 000
  - A: 440 → 310
- Last 2 characters:
  - 0: 0000 → 0000 (unchanged)
  - C: 4000 → 0000
  - B: 4400 → 2000
  - A: 4442 → 2400

Certain S and X groupers are also eligible for enhancements. Provided that $H \leq 3$, the enhancement transforms their EF to 24. Obviously this is not beneficial if the existing EF is 21, for example.

3 Key Attributes from Class and Group

I preface this section by stating that will base our calculations off of the enhanced version of any A/S-groupers, unless stated otherwise. We will also refer to the 14 character string as:

$$ABCD - EFGHI - JKL - MN - (A_2B_2C_2D_2E_2F_2G_2)$$

We also use Boolean logic.

3.1 Routine

Each person must go through a "Routine" twice a day. Some combination of the following aspects comprise a routine:

- Unhanced to Enhanced

  $$D \leq 1 + \Big((E_1 = 0 + F_1 = 0 + G_1 = 0)||(E_2 = 0 + F_2 = 0 + G_2 = 0)\Big)$$

- ChangeU

  $$(E \leq 2||F \leq 1) + G \leq 2 + (G \leq 1||H \leq 1) + I \leq 2$$
  $$\text{If X-Class}: D \leq 1$$

- ChangeL

  $$(E \leq 2||F \leq 1) + (G \leq 1||H \leq 1)$$

- Cat

  $$E \leq 1 + H \leq 2 + I \leq 3$$

- CatBag

  $$(E \leq 2||F \leq 1) + H \leq 2$$

- Support

  $$E \leq 2 + H \leq 1 + J = 0$$

- ChangeD

  $$(E \leq 2||F \leq 1) + (G \leq 1||H \leq 1) + H \leq 2$$

- Clean (using un-enhanced)

  $$\Big(A \leq 1 + B \leq 1 + C \leq 1 + D \leq 1\Big) \,||\, \Big((G \leq 1||H \leq 1) + (G \leq 2 + I \leq 2)\Big)$$

- Swing

  $$\Big(A \leq 1 + B \leq 1 + C \leq 1 + D \leq 1\Big) \,||\, \Big(F \leq 2 + G \leq 1 + H \leq 2\Big)$$
  $$\text{If X-Class}: D \leq 1$$

- OHP

  $$G \leq 2 + H \leq 3 + I \leq 2$$

- Manual

  $$(E \leq 2||F \leq 1) + G \leq 3 + H \leq 3 + I \leq 3$$

3.2 General Tasks

Each person also has general tasks they must complete, independently of their Routines. We use the enhanced version for calculations here.

When I say $A_x + C_x$, I mean that either $A_1 + C_1$ has to be true or $A_2 + C_2$ has to be true. $A_1 + C_2$ would not apply, as I assert that the x values are equal.

- Driver
  - Type 1:
    $$A_x \leq 1 + C_x \leq 3 + E_n \leq 2 + F_n \leq 2$$
  - Type 2:
    $$E \leq 2 + F \leq 2$$
  - Type 3:
    $$(E_x \leq 2 || F_x \leq 2) + J \leq 1$$
  - Type 4:
    $$J = 0 + L \leq 3$$
  - Type 5 (Not for XA or XC groups):
    $$J = 1 + L \leq 3$$
- Bread
  - If $L \leq 2$:
    $$(E \leq 2 || F \leq 1) + G \leq 3 + H \leq 3 + I \leq 3$$
    $$\text{OR } A = 0 + B = 0 + C = 0 + D \leq 1$$
  - If $L > 2$:
    $$(E \leq 2 || F \leq 1) + H \leq 2$$
- Fire
  $$E \leq 2 + H \leq 2 + I \leq 3$$
- FireMW
  $$(E \leq 2 || F \leq 3) + G \leq 3$$
- Water
  $$(E \leq 2 || F \leq 1) + G \leq 1 + I \leq 1$$

3.3 Chair Type

Recall from the introduction that "Chair" was one of the four modes. Chair is the mode in which people spend the most time. There are 4 types of Chairs: **None, Analog, Power, Shell**. The type is selected according to the following criteria:

- None:
$$B \leq 2 + C \leq 2 + D = 0$$
- Analog:
$$F \leq 2 + G \leq 1 + I \leq 1$$
- Power: All remaining EXCEPT XA and XC.
- Shell: All XA and XC.

4 Calculating Level from Routine and Tasks

We now introduce a powerful tool, the **level**. The decision tree is as follows:

- Is chair type None?
  - Level is 0.
- Does the person meet the criteria for every aspect of Routine?
  - Level is A.
- Out of the following aspects, which aspects does this person meet the criteria for?
- ChangeU, ChangeL, Clean, Swing, Manual, OHP
  -

5 Inputs and Commands

I introduced the concept of "Tech 1" in January 2021. Essentially, every person with the Power or Shell Chair type is endowed with a **hook**. Before we can discuss hooks, however, we must discuss **control schemes**.

5.1 Primary Input Commands

Primary inputs can come from several different sources. Below I lay out all the possible inputs at a tech level of 0, and the criteria for having these inputs available.

- LF Inputs
  - LF01 (LF Switch 1)
    $$A_1 \leq 3$$
  - LF02 (LF Switch 2)
    $$A_1 \leq 2$$
  - LF10 (LF Stick)
    $$A_1 \leq 1$$
- RF Inputs
  - Same as LF inputs, just with A_2 instead of A_1.
- LH Inputs
  - LH01 (LH Switch 1)
    $$E_1 \leq 3 || F_1 \leq 3$$
  - LH02 (LH Switch 2)
    $$E_1 \leq 2 || F_1 \leq 2$$
  - LH10 (LH Stick)
    $$\Big(E_1 \leq 2 \ + \ F_1 \leq 2\Big) \,||\, \Big(E_1 \leq 1\Big) \,||\, \Big(G_1 \leq 3 \ + \ (E \leq 2 || F \leq 1)\Big)$$
- RH Inputs, similarly to RF/LF, use 2 instead of 1.
- xH Inputs
  - These inputs can be either mapped to LH or RH, but not both.
  - xH21 (H Pen)
    $$\Big(E_x \leq 2 \,||\, F_x \leq 1\Big) \ + \ G_x \leq 3 \ + \ H_x \leq 2$$
  - xH31 (H Screen) - Assert class is not X, then:
    $$G_x \leq 3 \ + \ H_x \leq 3 \ + \ I_x \leq 3$$

  - xH41,42 (H LC, RC)

$$E_x \leq 3 \,||\, (G_x \leq 3 + H_x \leq 3)$$

- HD Inputs
  - HD00 (C Stick)

$$J \leq 1$$

  - HD11 (HB)

$$J \leq 2$$

  - HD12,13 (HBR)

$$\text{A and S Classes: } J \leq 3$$

$$\text{X Class: } J = 0$$

  - HD21,22 (H Switch 01, 02)

$$\text{A and S Classes: } J = 0$$

$$\text{X Class: } J \leq 3$$

- SP Inputs
  - SP01,02 (Sip s, p)

$$L \leq 3$$

  - SP03,04 (Sip S, P)

$$K = 0 + L \leq 3$$

  - SP11 (M Click)

$$L \leq 2$$

  - SP21 (Vox)

$$L \leq 1$$

  - SP31 (M Pen)

$$J \leq 2 + L \leq 2 + \text{Not XA / XC}$$

$$\text{OR } J \leq 2 + L = 0$$

- EE Inputs
  - EE01 (Absolute Pos)
  - EE11,12 (EE L, EE R)
  - EE21,22,23,24,25 (EE Sequence)
  - The above EE inputs are available to anyone, regardless of 14-string.
- Mx Inputs

  - We will go into more detail on Mx inputs in later sections. For now, disregard these. These will appear when the Tech Level is greater than or equal to 3.0.

Note that these inputs are only available for the modes Power Chair and Shell Chair. Other modes may have different control schemes.

We establish the following rules for primary inputs:

1. Two inputs in the same group cannot be input simultaneously. For example, HD21 and HD11 cannot be input simultaneously, while HD21 and EE01 can.
2. SP31 and HD inputs cannot be input simultaneously.
3. SD31 and EE inputs cannot be input simultaneously.
4. When SP31 is ready, SP01,02,03,04,11 are all available.
5. When xH21 is ready, xH01,02 are available.

Then, we establish that each input has certain variables associated with it.

- readyTime: The time it takes to go from the "unready" position to the "ready" position for this input.
- unreadyTime: The time it takes to go from the "ready" position to the "unready" position for this input.
- delaySelf: The time it takes before this same input can be hit consecutively.
- delayGroup: The time it takes before an input in the same subgroup can be input (same tens digit). For example, the time between inputs SP01 and SP03, but **not** SP01 and SP11.

Ready and Unready times apply only to SP31 and xH21. When these groups are in the "ready" position, all other inputs in the same group not expressly listed above as available are not available until the group is set back to "unready".

Please see the tech_primary.txt document for information on the above variables.

5.2 Input Information Conveyed

Here we discuss what information these inputs convey.

5.2.1 Unary

- These inputs do not have a time component. They are simply input.
  - SP21
  - EE21,22,23,24,25

5.2.2 Unary On-Off

- These inputs only convey whether or not the input is being pressed or not at a certain moment. They cannot be enhanced by being combined with other inputs in the same group.
  - LF01,02
  - RF01,02
  - LH01,02
  - RH01,02
  - xH41,42
  - SP01,02,03,04
  - SP11
  - EE11,12

5.2.3 Enhanced Unary

- These inputs can be combined with other inputs to create enhanced inputs.
  - HD12,13 + HD11
  - HD21,22 + HD11

5.2.4 Directional

- These inputs provide continuous X and Y values.
  - LF10
  - RF10
  - LH10
  - RH10
  - HD00
- Note that these inputs have "precision" values for r and θ. For example, the default HD00 has precision values of $r = .5$ and $\theta = 45$, indicating that r can take on the values $0, 0.5, 1$, and θ can take on any multiple of 45 degrees. (I would like to use π but I don't trust floating point math).

5.2.5 Absolute Directional

- These inputs are similar to Directional, but their value can be changed while violating Intermediate Value Theorem.
- For example, using HD01,02,03,04, moving from (1,1) to (-1,-1) would require me to pass through X=0 and Y=0. With an Absolute Directional input, however, I can move from (1,1) to (-1,-1) instantly.

  - xH21
  - xH31
  - SP31
  - EE01

- Again, these inputs have precision values, though instead of r and θ they are x and y. Assuming the area of input is $(1, 1)$, a precision of $x = 10$ and $y = 5$ indicates that 50 total areas on the screen are available to be selected.

- As such, we can use Absolute Directional commands to define new Unary commands.

We will explore this later, but each person has a **latency** value associated with them. This determines the "error value" of their input durations and precisions, so to speak. For example, if a person's latency value determines that their input duration can be at most 0.2 seconds precise, and they have that input mapped to a rotation to where they need to be precise to within 10 degrees, then the **velocity** can be at most 1/7.2.

5.3 Secondary Commands

Secondary commands are where the "useful" stuff happens. We use primary inputs to map to secondary commands, in order to make our person do something. **The following information applies only to Power and Shell Chair modes**.

5.3.1 Universal Commands

We have certain **Universal Commands**, where no matter what sub-mode we are in, the same primary inputs will map to the same secondary commands. They are as follows:

- SWITCH: Needs Unary input
- STOP: Needs Unary input
- EMERGENCY: Needs Unary input
- NEXT / TOGGLE: Unary input, takes up to 2.
- LEFT CLICK: Unary input
- RIGHT CLICK: Unary input
- BACK: Unary input

In total, 7 or 8 unary inputs will be reserved for Universal Commands. Left Click and Right Click themselves can be mapped to secondary commands in any sub-mode, while SELECT / TOGGLE can be mapped in only certain sub-modes.

In all subsequent sub-modes, each input-command mapping is only valid in that sub-mode. Additionally, there are sub-submodes we refer to as **axes**, switchable via the SELECT / TOGGLE commands, where input-command mappings can be made valid in only that axis and sub-mode.

We refer to a **2-switch** as a group of two associated unary on-off inputs in the same group: for example, LH01 and LH02. A **1-switch** is a single unary on-off input.

5.3.2 Sub-Mode: Drive

The default mode. Inputs needed:

- Precision of 15 degrees and 0.5 units/s (range -1 to 1) desired. Must be able to reach 0 units/s within 1 second at all times.
- Forwards / Backwards
- Left / Right
- Preferred Scheme: Directional input mapped to both.
- Scheme 2: Two 2-switches mapped.
- Scheme 3: One 2-switch, using TOGGLE to switch.
- Scheme 4: Two unary inputs, using TOGGLE to switch. Each input is fixed to +/- 0.5 units/s or +/- 15 degrees.

5.3.3 Sub-Mode: CPOS

- Base Up / Down: Precision 0.25 units (0 to 1).
- Base Back / Forwards: Precision 10 degrees (0 to 90).
- H Theta Up / Down: Precision 15 degrees (0 to 90).
- Knee Theta Up / Down: Precision 30 degrees (0 to 90).
- Casters Raise / Lower: No precision necessary.
- Preferred: 5 2-switches.
- Scheme 2: 4 2-switches and a unary input for Casters Raise / Lower.
- Scheme 3: 2 2-switches, a unary input for CR/L and TOGGLE (2 axes).

- Scheme 4: 1 2-switch, a unary input for CR/L, and TOGGLE (4 axes).
- Scheme 5: 1 unary input, with each input moving by the precision amount. TOGGLE (10 axes).

5.3.4 Sub-Mode: Camera

- Toggle Camera: Mapped to TOGGLE.
- Rotate Camera CW/CCW. Precision 30 degrees (0 to 360).
- Rotate Camera +Z/-Z. Precision 15 degrees (0 to 180).
- Camera Zoom +/-. Precision 0.5x (1x to 3x).
- Unfortunately, we cannot remap TOGGLE in this mode. Thus the above inputs are according to priority of how many unary inputs are available.
- Note that CCW and -Z are not strictly necessary, since we have 360 degree cameras, and CCW and -Z can simply be held down for however long until it loops back around.
- Preferred: 3 2-switches.
- Scheme 2: 2 1-switches and a 2-switch.
- Scheme 3: 2 2-switches.
- Scheme 4: 1 2-switch (we lose zoom).
- Scheme 5: 1 1-switch (we lose +Z).

5.3.5 Sub-Mode: Text Input

- Select Preset Folder: LEFT CLICK
- Select Preset Word/Phrase Within Folder
- Cursor Position (On-Screen Keyboard Entry)
- Select Object at Cursor Position: LEFT CLICK
- Quick Backspace 1 Character
- Quick Backspace 1 Word
- Quick Backspace 1 Sentence
- Return Current Phrase
- Clear Everything: BACK

- One of the Absolute Directional inputs will be mapped to Cursor Position. Then, 4 unary inputs will be mapped to the quick backspaces and "Return Current Phrase".

It is generally preferred to have minimal overlap between the inputs used in the sub-modes, as then multiple commands can be issued without having to switch modes, giving the user much more flexibility.

5.4 Hooks

We now shift our focus to hooks. This is an additional sub-mode, unlocked at tech level 1. The hook consists of two metal rods of length 3 feet each, connected by a ball-and-socket joint. At the end of the second rod is a gripper that can hold objects up to a reasonable size and weight. By default, the hook will be controlled through the camera; thus the hook will likely inherit many of the camera sub-mode's arguments.

The inputs are as follows, for **Hook 1**:

- Move Hook Forwards / Backwards (towards Camera Crosshair): Precision 0.5 inches.
- Rotate Crosshair CW / CCW: Precision 5 degrees (0 to 360).
- Rotate Crosshair +Z / -Z: Precision 5 degrees (0 to 360).
- Grip / Release Object: Unary input.
- Rotate Head CW / CCW: Precision 10 degrees.
- Scheme 1: 4 2-switches and a unary input.
- Scheme 2: 2 2-switches and a unary input. 2xTOGGLE axes.
- Scheme 3: 1 2-switch and a unary input. 4xTOGGLE axes.
- If Scheme 3 cannot be provided, the user does not receive Hook 1.

6 Secondaries and Tech Level 2