Block Description (Basic Function)

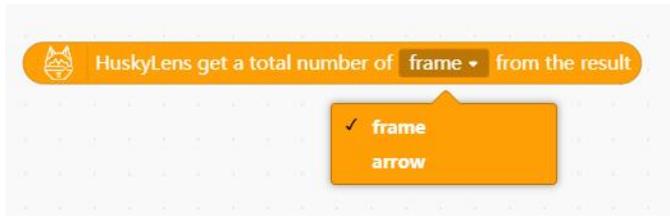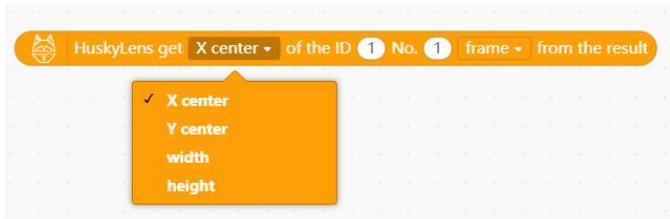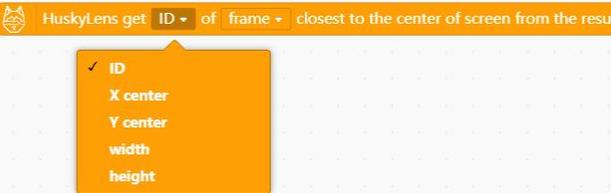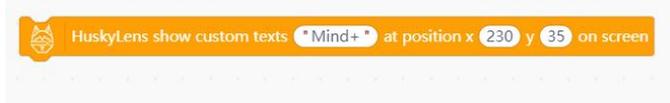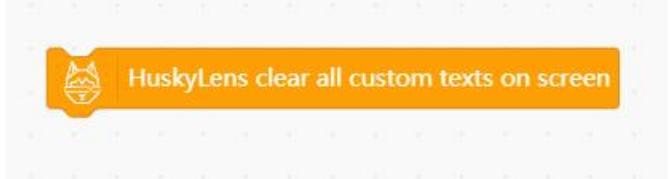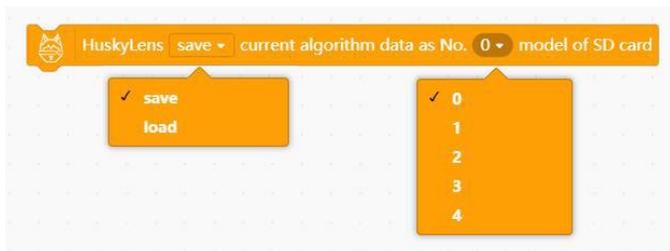| Block	Description
HuskyLens initialize pin ⚙ until success Communication Method: I2C Address: 0x32 SDA(Green): P20 SCL(Blue): P19	This command block is used for initializing program and only needs to be executed once. It's placed between the beginning of main program and looping execution. Users can select I2C or serial for communication without changing I2C address. **Please note that the "output protocol" of HuskyLens should be set to be consistent with the program, otherwise data couldn't be read.**
HuskyLens switch algorithm to Face recognition ▾ ✓ Face recognition Object tracking Object recognition Line tracking Color recognition Tag recognition Object classification QR recognition (Edu only) Barcode recognition (Edu only)	This command block is used to switch algorithms. Users can switch to other algorithms anytime, but only one can be executed at a time. Please note that it takes time to switch algorithms.
HuskyLens request data once and save into the result	Request data once from HuskyLens and save it into the "result" (stored in the memory variable of the main board, and when requested, the data in the memory will also be refreshed once), and then the data can be obtained from the "result". The latest data can be obtained from the result only when this block is called.
HuskyLens check if ID 1 is learned from the result?	Check if the IDx has been learned from the requested "Result".
HuskyLens check if ID 1 frame ▾ is on screen from the result? ✓ frame arrow	Check if the IDx is on screen from the requested "Result". The frame refers to the algorithm when the object on screen is frame, while the arrow corresponds to the algorithm when it's arrow. Select arrow when the current algorithm is line-tracking.For others, select frame.
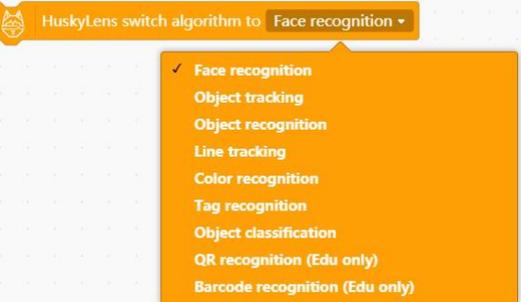
HuskyLens get **X center ▾** of ID **1** frame ▾ from the result ✓ X center Y center width height	Get the parameter of IDx from the requested "result"; return -1 if this ID is not on screen or unlearned.
HuskyLens get **X beginning ▾** of ID **1** arrow ▾ from the result ✓ X beginning Y beginning X endpoint Y endpoint	Get the parameter IDx from the requested "result"; return -1 if this ID is not on screen or unlearned. It corresponds to the output result of line-tracking algorithm.
HuskyLens get a total number of learned IDs from the result	Get the total number of learned IDs in the current algorithm from the requested "result". Long press the select button of HuskyLens to open advanced settings, then set whether to learn multiple IDs.
HuskyLens get a total number of **frame ▾** from the result ✓ frame arrow	Get the total number of "frame" or "arrow" on screen in the current algorithm from the requested "result", including the learned and unlearned objects.
HuskyLens get a total number of ID **1** frame ▾ from the result ✓ frame arrow	Get the number of objects with the same ID in the current algorithm from the requested "result". For example, if two face photos of a person are in view, the number is 2.
HuskyLens get **X center ▾** of the ID **1** No. **1** frame ▾ from the result ✓ X center Y center width height	Get the parameter of the y_{th} frame of IDx in the current algorithm from the requested "result". Multiple objects may share the same ID (for example, two photos of a person appear on screen at the same time), please note that objects with the same ID are in random order.
HuskyLens get **X beginning ▾** of the ID **1** No. **1** arrow ▾ from the result ✓ X beginning Y beginning X endpoint Y endpoint	Same as the one above; the difference is that this one reads the data of arrows.

HuskyLens check if `frame ▾` is on screen from the result? ✓ frame arrow	Check if there are frames or arrows on screen from the requested "result", including the learned (ID>0) or unlearned; return 1 if there is one or more.
HuskyLens get `ID ▾` of `frame ▾` closest to the center of screen from the result ✓ ID X center Y center width height	Get the parameter of the frame closest to the center of screen from the requested "result", and ID of the unlearned frame is 0; return -1 if there is no frame.
HuskyLens get `ID ▾` of `arrow ▾` closest to the center of screen from the result ✓ ID X beginning Y beginning X endpoint Y endpoint	Same as the one above; the difference is that this one reads the data of arrows.
HuskyLens get `ID ▾` of the No. `1` `frame ▾` from the result ✓ ID X center Y center width height	Get the parameter of the N_{th} (random order) frame on screen from the requested "result", including the learned (ID>0) and unlearned objects.
HuskyLens get `ID ▾` of the No. `1` `arrow ▾` from the result ✓ ID X beginning Y beginning X endpoint Y endpoint	Same as the one above; the difference is that this one reads the data of arrows.
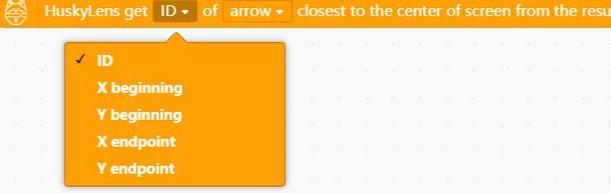
* The screenshots above are taken from Mind+V1.7.0 RC3.0.

New Blocks in Mind+ V1.6.4 or Later

Block	Description
HuskyLens learn ID **1** once automatically	This block allows to program HuskyLens to learn, equaling to the **Learn Button;** when it's called once, HuskyLens learns once, and users can run it repeatedly for multiple learning.
HuskyLens forget all learning data of the current algorithm	This block is used to clear all the learned content in the current algorithm, equaling to **forget** function.
HuskyLens name ID **1** of the current algorithm as **"Mind+"**	This block is used to name an ID; the ID name of different algorithms can be set separately.
HuskyLens show custom texts **"Mind+"** at position x **230** y **35** on screen	This block is used to display texts at any place on screen. Only English letters, numbers and symbols are supported.
HuskyLens clear all custom texts on screen	This block is used to clear all **custom texts** on screen.
HuskyLens take **photo ▾** and save to SD card ✓ photo screenshot	This block is used to take photo or screenshot and save it into the TF card in HuskyLens. Remember to insert a TF card before calling it. **Screenshots** contain contents on the screen like texts and frames, while **photos** contain only the image formed in the camera, and no texts or frames.
HuskyLens **save ▾** current algorithm data as No. **0 ▾** model of SD card ✓ save / load ✓ 0 / 1 / 2 / 3 / 4	This block is used to load or save the model to the TF card in HuskyLens. Remember to insert a TF card before calling it. 5 sets of data can be stored, and the data of each algorithm is saved separately.