CREATION: CONSTRUCTION-CYCLE 2
[ALI NAJIB] CORE FILE: CREATION: CONSTRUCTION-CYCLE 2

Project Progress-Essay, Research Proposal. (Formalize later)

Substance first. Rewrite later. Convolutional Neural Network construction by Deep Reinforcement Learning and Chip-placement by Deep Reinforcement Learning are one-to-one translatable. Setting-dependency lies in variation by specialisation of observer cognition. AlphaFold's architecture serves as one example for a specialized observer. Graphcast is another example. AlphaFold can be AlphaChip-assembled, given the present elements in AlphaFold. AlphaTensor and AlphaDev serve as constructors, and both have indeed been written in the framework of Deep Reinforcement Learning. The same holds for MuZero. The crux lies in the architecture of observation and its transformability. The elements of the architecture are setting-dependent, and the congnition of the observer varies by specialization.

Notably, two-dimensional convolutional neural-network construction and chip placement with Deep Reinforcement Learning are one-to-one translatable, and for Navigation Control can we use a secondary, iteratively transformable, neural-network that takes inputs on iteratively transforming primary neural-networks. The essence of any form of application lies in their respective domain-constants, translatable to learning architecture. (Examples are to be found in Hubble's Law's cosmological constants for pre-filtering red-shift and blue-shift as utilized in the LOFAR-Sky Survey.)

Obstacle detection and circumvention in motion through space of observables is computable and simulable in the framework of Deep Reinforcement Learning. Deep Sea deployment of Deep Sea Mining Vehicle is reconstructible in traversal over every other platform if given the required sensors. (In DSMV: $\delta_{\text{tar}}, \delta_{OLS}, i_{RI}, \dot{w}_b$, including distance to obstacle). DSMV target path Navigation Control Learnable for DSMV realization, and in turn Drone realization, or Land Crawler, Mars Rover realization.

Path-planning in obstacle circumvention feasible by incorporation of setting specific obstacle-type in pseudospectral control for unknown ocean currents, generalizable to air currents. Navigation control manual construction potential starting point in path planning for underactuated vehicle or roamer by sensor-induced state-representation restricted to vision captured by installed visual system. (Complication in state-representation must correspond to complexity in area of deployment.)

Encoded observable information extraction from information-technology systems such as HIS, LIS, PACS, EMR, existing monitoring for utility traces. Mass: virtuality/reality synchronization for simulation pre-deploy. Examples for virtual spaces of observables include Financial Time Series Databases (Large Scale Cyberspace). Neuro-image by IC-scan in higher-dimensions. Traversal over Internet of Things, all traversable by method of traversal in physical space, by translation brain-network topology representation in structure, and dynamics allows for a learner's motion, or traversal, over the brain, in pursuit, e.g., of an appropriate Network hyper-parametrization (as to fit the network's orgin in biological cognition). Motion on a guided path would e.g. occur by deep reinforcement Learning. This, indeed, lays bears the possiblity of mathematical optimization over all of a neural-network's parameters, leaving no room for human intuition in tweaking hyperparameters. Applications in financial time-series owe their origin in databases on elementary trade. **Delayed addition.** Correctness check and efficiency consideration by fixing constant nature, such as constant Hubble's Law. For focus of learning on the unknown,

comparable to temporal basis for time-series. Detection and capture vision dependent on nature detectee and translatable to architecture learning. Navigation through observables may be translated to plain terms as 'knowing where to look over all things that can be seen', 'zooming in', 'varying telescope orientation', 'shifting microscope sample'. Navigation through observables is an observable learnable by, e.g., RNN constructed by Chip placement. Note that navigation through observables assumes inherent limitation in range observer vision. Necessity of navigation through observables assumes excessive abundance of observables for preconceptualization by observation of one vicinity. Autonomous vehicle localization and map-matching serve to bear hints towards heuristic boundaries for navigation towards, or over, preconceptualization. (Momentary example is based on extent of conceptualization, and differences therein over continuous points in space of observables). Higher-orbit or deep-space or deep-sea (Space/Ocean)craft navigation policy towards guided exploration or faring towards body, based on X-ray pulsar capture, may bear hints towards dynamics of observer over space of observables, the latter yielding a similar extent of mystery, and encoding after observation, by conceptualization.

First Preliminary Network Architectural Design. Sources include DeepMind's AlphaFold, AplhaStar, AlphaChip, AlphaGlobe, AlphaDev, AlphaMolstruct, AlphaProteome, Graphcast. The below outlines a blueprint of Network Architectural Design.

$$\overset{**}{\boxed{\text{Reality}}} \overset{(1)}{\leftrightarrows} \underset{\cdots}{\boxed{\text{cross-DL}}} \overset{(2)}{\leftrightarrows} \underset{\cdots}{\boxed{\text{Learner-calibration DRL}}} \overset{(3)}{\leftrightarrows} \underset{\cdots}{\boxed{\begin{array}{c}\text{Scoper of}\\ \text{Parametrization DRL}\end{array}}} \overset{(3)}{\leftrightarrows} \cdots$$

$$\boxed{\text{(simulated) World}} \overset{(4)}{\leftrightarrows} \overset{**}{\boxed{\text{Walker DRL (†)}}} \overset{(5)}{\leftrightarrows} \boxed{\text{Walker-calibration DRL}} \overset{(6)}{\leftrightarrows} \cdots.$$

A special case of the frame

$$\boxed{\text{Reality}} \overset{(7)}{\leftrightarrows} \boxed{\text{DGN}}.$$

All are interconnected.

$$\begin{array}{ccc} & \overset{***}{\boxed{\text{Reality}}} \overset{(8)}{\leftrightarrows} \overset{****}{\boxed{\text{Cross-DL}}} & \\ \overset{***}{\boxed{\begin{array}{c}\text{Globe Maps}\\ \text{PcChips}\\ \text{MolStruct}\\ \text{Conduct Cross-observation}\end{array}}} & \overset{(9)}{\leftrightarrows} & \overset{****}{\boxed{\begin{array}{c}\text{Graphcast}\\ \text{AlphaChip}\\ \text{AlphaProteome}\\ \text{CrossDL}\end{array}}} \end{array}$$

(Sources include DeepMind's AlphaFolds, AlphaStar, AlphaChip, AlphaGlobe, AlphaDev, AlphaMolstruct, AlphaProteome, Graphcast. Arrow directed up and down, to be integrated, indicate extension of representation and correction higher degrees of extensivity). For all the other symbols follows now a legend.

First Preliminary Network Architectural Design. (Legend One)

1. →: interpretation, ←: representation.

2. →: interpretation, ←: representation. *AlphaChip network-construction defines this link.*

3. →: interpretation, ←: representation. *AlphaChip network-construction defines this link.*

4. →: interpretation, ←: navigation. *AlphaChip network-construction defines this link.*

5. →: interpretation, ←: representation.

6. →: inspection, ←: correction. *AlphaChip network-construction defines this link.*

7. →: inspection, ←: correction. *AlphaChip network-construction defines this link.*

8. →: interpretation, ←: representation.

(**). The two boxes marked by (**) are interlinked by navigation of [Walker-DRL] over [Reality] and interpretation of [Reality] by [Walker-DRL].
(***). The two boxes marked by (***) are interlinked by representation of [Reality] by [GlobeMaps, PcChips, MolStruct, ConductCrossObservation].
(****). The two boxes marked by (****) are interlinked by representation of [CrossDL] by [GraphCast, AlphaChip, AlphaProteome, CrossDL].

First Algorithmic Set-up. [First Algorithmic Set-up; First expression in terms of popular notions]

1. Conv3D network assembly in stages defined by distance between network specifications with assembly procedure by AlphaChip chip-placement DRL. Implementation follows from a specification of all the attributes adhering to a DRL-agent that performs network-construction. Demonstration follows from AlphaChip.

2. Setting-specificity might bring rise to inputs that are no outputs of vision. Network assembly procedure possible by Chip-placement DRL if distances are defined via setting specificity. (Difficult example is AlphaFold assembled by AlphaChip).

3. Conv3D Network Chip placement assembly possible by calibrating on one time-varying variable, see (1). For (online) deployment and assembly of Conv3D Network, attach Conv3D Network to RNN-Network as, e.g., specified in [DLB], Figure 10.4.2.

4. Deep Reinforcement Learning: Area of deployment, either in virtual or physical space, e.g. IoT or historical records, or some region. Virtual space can simulate physical space (use, e.g. Unity or Unreal Engine or SimuLab).

   States may be defined by oculary picture(s), or oculary location and orientation, and actions may be defined by oculary-movement. Rewards are derived from Algorithm (3) installed observer.

Training may be performed by PPO or SARSA(λ) both being online reinforcement learning algorithms. Training by performing any offline reinforcement learning algorithm will take much time before convergence due to space's breadth.

5. Hyperparametric consideration: area of deployment must be known for definition states under algorithm 4, this is true for simulated space, otherwise area must be downloaded by e.g. agent's random walk, as initialization stage, or a new reinforcement learning algorithm. Under Algorithm 3, value for variable in calibration is set under oculary's initial position, e.g., or is searched by orientation, and choice of oculary influences output algorithm 1.

Pinpointing the problems in existing techniques in Machine Learning. Consider $\mathcal{D}_f$, residing in an environment and bearing phenomena about which one might have some preconception, but little notion of theory towards a representable property of interest. A first problem is to find the representation, and this might in turn occur by a first application of [CrossDL] $\rightleftharpoons$ [WalkerDRL] equipped only with Vision in an area near $\mathcal{D}_f$, either in a non-virtual or virtual sense. Afterwards, equipped with representations of $\mathcal{D}_f$, one may re-deploy the device for learnability of $\mathcal{D}_f$. Stage 1 deployment occurs by observation of the phenomena in a sense that need not yield more clarification than initial encounter, i.e. first contact, towards $\mathcal{D}_f$, that at this point might be known by any data type, e.g. that which made the researcher declare initial encounter by Vision.

If the representation resulting from the initial Vision encounter succeeds in representing the property of interest, the encounter encoded by $\mathcal{D}_f$ may be used for stage 2 deployment, in which case stage 1 would result in the representation $\mathcal{D}_f$ for encounter $\mathcal{D}_f$

Learning may occur online or offline, and the assumption of data availability is essential. One has some preconception of the relation between target and relevant features, and these features have been selected beforehand therefore. Any preconception may occur on the basis of premonition or established theory. Learning the environment happens thereafter, on the basis of the latter isolated environment.

Representing any target, or feature, or environment (One). Many problems in conventional machine learning seek to learn for predictability, or explainability. There are phenomena that, however, may barely be observable, or encodings that may cause information loss to great extents. One may not observe much latency in a defined numeric feature that does not in the least represent $\mathcal{D}_f$, the encounter of interest.

To learn an environment is to enable accurate generation of extensions of the environment, in the broadest sense that an environment reveals itself. The matter of feature-selection occurs itself by learning, and so does the matter of feature-identification occur itself by learning (from vision-lens encoded input). The same would hold for any target, whence more information than mere binary encoding represents an encounter of interest.

Scale One. It is reasonable to assume that data-entanglement with $\mathcal{D}_f$ occurs near $\mathcal{D}_f$, and while this is of interest for small-scale, conventional learning, one may be interested in relations that exist over large distances in space, either temporal,

physical or virtual. One far-fetched device that would learn and capture such an entanglement is a satellite equipped with [CrossDL] for relations that attract [WalkerDRL]. The idea, again is to specify an inquiry from which some representable target of interest follows, to be encoded in an alternative reality to which we have access, so that the representable target may be reconstructed and extended. The target may be the means by which first encounter has occurred, and may entail an entire physical environment. The device, be it an orbital satellite or a drone that explores dataspace by traversing planetary orbit and performing areal rend-a-vouz respectively. As one would seek preconception on a target, would the device traverse an area, or a zone, to mine for data-entanglement. The problem resides mainly in feature-identification from, e.g., vision by lens, and to define the states in some state-space so as to optimize environment generation. In summary, the device is the learner as a whole, a completion. We construct and configure the device by specifying the objective, the target. The device subsequently observes, learns, observes, learns, ..., until reached terminal criterion.

Feature-collection and identification One. *State-specification.* Pattern happens by preconception of environment of deployment to be automated by [WalkerDRL]. Feature-collection depends on device of observation, thermolens given spectral features that might relate to $\mathcal{D}_f$, and thermolens might be captured by HDR lens. Feature-collection by VisionCNN captures many features that are likely entangled with $\mathcal{D}_f$, captured by only a Vision-lens. (The sense of 'lens' is highly general: for a start can lenses be virtual data-collectors over some virtual DataSpace). Feature-collection depends on device of observation. Feature-collection is possible via indicator-measurement, as, e.g., through a thermostat, or a level of chemical matter extraction as measured by pH. Again, both the latter phenomena may be captured by a Vision-lens. Feature-Collection in sound may be processed by CNN by sound-representation in audiogram. Other ways of incorporation towards state s are to be considered for reasons of efficiency. (This form of data might rarely be useful.) The problem of reward-specification reads as $[\text{CrossDL}]_{\mathcal{D}_f}(\mathcal{D}_{\text{data}}) \overset{L'}{\to} (L_{(i_1,\ldots,i_n)})_{\text{Combper}(i_1,\ldots,i_n)}$. For momentary convenient simplicity, consider the problem $[\text{CrossDL}]_{\mathcal{D}_f} \overset{L}{\to} (L_{ij})_{i,j\in\{1,\ldots,n\}}$. The function L is defined and results in $(L_{ij})_{i,j\in\{1n\ldots,n\}}$ with magnitudes depending on input $[\text{CrossDL}]_{\mathcal{D}_f}(\mathcal{D}_{\text{data}})$. The component $[\text{CrossDL}]_{\mathcal{D}_f}$ depends on finding good representations of $\mathcal{D}_f$. Specification of $\mathcal{D}_f$ is essential, and predictability, or computability, or learnability depends on this representation.

First Preliminary Elementary Specification. The below have been specified under the assumption of the reader's slight familiarity with reinforcement learning. We may invoke references for bringing full algorithmic detail, laying waste to all dependency in pre-defined function. The reader might derive great comfort from mathematical proof distilled to first principles, and so shall we provide some. Formulation [WalkerDRL] in full extension. Difficulties lie in environment-specification in terms of the space defined by state-action-reward tuples (s, a, r).

First Preliminary Algorithm. First Preliminary Algorithm.

Second Preliminary Algorithm. Second Preliminary Algorithm.

Algorithm 1: First Preliminary Algorithm.

$(s', a', r') \leftarrow$ Environment Learning: Exploring Starts (e.g.)
$(\pi_A, \pi_C) \leftarrow$ Navigation Learning: PPO (e.g.)
Gen. $\leftarrow$ Generation$(\pi_A, \pi_C, [\text{CrossDL}]_{\mathcal{D}_f})$ $\square$

Algorithm 2: Second Preliminary Algorithm.

$(s^2, a^2, r^2) \leftarrow$ Environment Learning: Exploring starts over$\{s^2, a^2\}$
Gen. $\leftarrow$ Generation$(\pi, [CrossDL]_{\mathcal{D}_f})$
Gen. $\leftarrow$ Generation$(\pi_A, \pi_C, [\text{CrossDL}]_{\mathcal{D}_f}))$
Re-define $\{s^2, a^2, r^2\}$ and vary over loss-configuration. $\square$

Third Preliminary Algorithm. Reinforcement Learning Configuror with state-space (s^3, a^3, loss-configuration(r)), first lower-order reinforcement learner hyper-parameters, first lower-order reinforcement learner action space). One such a first lower-order learner by reinforcement is [NavigatorDRL]. To learn the proper configuration of [NavigationDRL] deploy the following and iterate until threshold criterion or (near) guaranteed eventual convergence.

Algorithm 3: Third Preliminary Algorithm.

Environment learning: $\{s, a, r\} \overset{\text{output}}{\leftarrow}$
Exploring starts over $\{s, a, \text{loss-configuration}(r, L)\}$.
Navigation Learning: $\{\pi\} \leftarrow$ PPO e.g.
Target Generation $\leftarrow$ Gen.
$\{s, a, r\} \leftarrow$ Environment Learning: ExploringStarts (e.g.)
$(\pi_A, \pi_C) \leftarrow$ Navigation Learning: PPO (e.g.)
Gen. $\leftarrow$ Generation$(\pi_A, \pi_C, [\text{CrossDL}]_{\mathcal{D}_f})$. $\square$

One elaboration on CNN-encoding of vision observations. Consider the testing for a learner by reinforcement. State start s_0: observation $\mathcal{D}$. Proceed, e.g., as follows.

$$\begin{aligned}
\mathcal{D} &\overset{\text{obs-CNN}}{\mapsto} (\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_{\text{k}}) \\
&\overset{\text{obs.-CNN}}{\mapsto} (\mathcal{D}_{11}, \ldots, \mathcal{D}_{1k}, \ldots, \mathcal{D}_{k1}, \ldots, \mathcal{D}_{kk}) \\
&\ldots \\
&\overset{\text{obs.-CNN}}{\mapsto} \text{"CombPer"}((\mathcal{D}_{i_1,\ldots,i_n})_{i_1,\ldots,i_n \in \{1,\ldots n\}}) \\
&\overset{L}{\mapsto} \text{"CombPer"}((L_{i_1,\ldots,i_n})_{i_1,\ldots,i_n \in \{1,\ldots n\}}).
\end{aligned}$$

Note that $\mathcal{D}$ is indexable to an order that is exponential to the amount of applications of obs-CNN to observation $\mathcal{D}$. For a representation of states for, e.g.

[NavigationDRL], denote therefore

$$
\begin{aligned}
s_0 &= (\mathcal{D}_{m_i})_{i=1}^{r} \\
s_1 &= (\mathcal{D}_{l_i})_{i=1}^{r} \\
s_2 &= (\mathcal{D}_{k_i})_{i=1}^{r} \\
&\dots \\
\hat{r}_i &= L_{s_i} - L_{s_{i-1}}, \quad i \geq 1 \\
\tilde{r}_i &= -\hat{r}_i, \quad i \geq 1 \\
r_i &= \{\hat{r}_i, \tilde{r}_i\}_{i \geq 1}, \quad i \geq 1 \quad \text{Simplest loss-configuration.}
\end{aligned}
$$

For positive r, seek optimal loss-configuration, via, e.g. an agent-configuror (see deployment 1.3 for an example hereof). Actions bring transition between states, practical examples of which can be found in drone-rotation, or movement of a virtual entity such as BipedalWalker-v2). Note that, in this simple setting, actions only allow transition between states that are distanced in index by 1. Transition probabilities $p(a|s)$ defined by noise and imperfection in observer and environment may be approximated by initial environment exploration. Policy $\pi_\theta(a|s)$ to be learned by, e.g., DQN. At terminal, settled state, reconsider all kernels (circularities allowed and encouraged). Indication for completion exploration phase may lie in a detected circularity in random traversal. The above example serves to illustrate, and is indeed most simple.

The storage of data by vision is highly general; consider the encoding of sound in audiospectograms, and the encoding of temperature via InfraRed lenses. The extent of preconceptualization in vision is measured by the scope of multi-channel CNN-application, therefore continue the procedure of data-point-entanglement until pre-defined threshold reached. Vision-adequacy. Assurance of threshold crossing by movement over environment of deployment can be granted by movement over environment, to give encodings for state-space $\mathcal{S}$ by and for [WalkerDRL]. Feature-collection with provided trace, a special case of vision-adequacy. Feature-collection with considered history of feature observation for capture of feature-movement.

The architecture of the vision CNN-Encoder constructor is a derivative of AlphaChip Policy Network Architecture. One might inquire into the nature of such a constructor, and one constructor of the greatest simplicity may be formulated as follows. An alternative to AlphaChip, of the greatest simplicity: CNN-Encoder construction by greedy feature-identifier selection. Feature-identification by CNN-Encoding with gradual addition of convolutional layers of dimension $C_i \times C_o \times K \times K$, where K defines the kernel size for specificity in feature identification, C_i is the input-channel size (depends on vision capturer via depth/color-encoding etc.) and C_o is defined by the out-channel size (sets the count of features to be considered in loss-determination and environment generation).

Up-/Down-sampling by (un)pooling capturable through specification K and C_o. Selection by input-type of K, C_i, C_o. One selection of an algorithmic nature reads as follows. For subsequent layers, reiterate the above algorithm with input-dimensions equal to the back-layer of the convolutional neural network to be expanded. One might define another overarching algorithm for a search of the optimal amount of layers with corresponding optimal layer-dimensions and network-propagation properties. A brute-force approach such as taken in the above algorithm can however come with excessive computational cost. One other approach lies in a setting of con-

structing the Conv2D network by Chip Placement by Deep Reinforcement Learning, see AlphaChip.

Readily Constructible Absolute Minimum One. The following does skip over many details. *Design of absolute minimality. [CrossDL]* contains only DNN, [*WalkerDRL*] defined by $\mathcal{S}$, pixel-value-set of vision-input of part of environment, $\mathcal{A}$, device movement over $\mathcal{S}$, $\mathcal{R}$, Loss-configuration over losses, $p(a|s)$: device-dependent, to be computed by explorative simulation (e.g. by MC) over environment. On-policy learned $\pi(a|s)$: traversal route over $\mathcal{S}$. Target specification: long measurement by Vision prolonged observation by measuring device in portable oculary. Time-interval between movements tweakable, e.g. 4 seconds to 4 days depending on activity-intensity over $\mathcal{S}$. Local optimum if any action results in negative reward, after which settle for optimal optimum.

Complication in transitions $p(a|s)$. State-space $\mathcal{S}$ may also be defined by location, or orientation, whence only rewards are derived from Vision observation. Stochasticity in $p(a|s)$ stems from device imperfection or action stochasticity, although determinism in $p(a|s)$ is no obstacle.

SoftwareReady. Improvements are possible in definition s in $\mathcal{S}$, in feature-consideration for r in $\mathcal{R}$, in device of observation and deployment (Drone better than non-mobile device) and a in A, in target representation, [crossDL] configuror component addition (configurors for optimization over improvement dimensions) (RL-optimization). For an extension, consider state-space mixtures in $\mathcal{S}$.

First Data-space Set-up. Deployment of algorithms are dependent on synchronization in index, by synchronization in time-indices and by synchronization in location-indices, a combination of indices of varying natures that have been synchronized, or new specification. Time-delay poses no obstacle to synchronization over indices that include a time-index. The synchronization exists between input and output realizations, and histories may be used since chronology bears more importance to functionality.

For deployment by simulated oculary in virtual space, data-capture is near automatic. For capture of data for dataspaces that might be more esotoric, first define indices over which to reset synchronization, and proceed with the above algorithms. In simulated space, all data-structure is enforcable by built-in programming. In spaces of data that have been constructed, dataspace distance is first defined, and for synchronization over indices other than time does one need a sense of order over the indices more than enumeration.

Evolution in output due to evolution in corresponding realizations can occur in various ways. Movement of the evolution can occur by an oculary's movement, or landscape's movement, or the activity of an EEG. (It is) It is the evolution in its entirety that must be defined. The same holds for the input, although an agent's movement itself captures part of the evolution (the agent's necessity follows the unobservability of one dataspace in its completion. Hence, a similar agent might be deployed over output dataspace.)

Conceptualization in its many forms would gain improvement upon completed simulation and obtained elements of defect. The RL-lens-sharpener-route of conceptualization is one framework through which to implement conceptualization and tweaking its specificity may give improvement, or a translation of another implemen-

tation. The RL-route gives leeway for continuous or discrete, (pre-learned) iterative sharpening, an attribute that is not shared with manual initialization or basic grid-search automated within time-intervals.

In the current implement, the lens sharpness is adjusted by a learner that has learned or is learning the optimal adjustment of lenses. The observer's navigation protocol has been pre-learned or is being learned by the exploring observing. This is the first implementation, most general.

Exhibitability One. An observer's interpreter drives the program. Thus seek for many interpreters. Envision interpreter for every setting over which knowledge retrieval might be desired, or needed for furtherment. Data can take many forms, thus would a setting-encoding network be better than interpretation after encoding? What other ways of traveling over a setting exist? A reinforcement learner has in its core built-in the policy, equippable by an observer. Would observation over a setting in its completion be necessary in some situations due to a traveler's restriction? How would such an activity occur? Worlds are big indeed.

Lens construction occurs by layer-after-layer assembly of a convolutional neural network according to a trained reinforcement learning agent that steers the type and specification and depth of each layer in the network. And this has shown effective in the chip-placement analogy. A similar demonstration would have to show the same effectivity for lens construction. What other ways of observing might there exist? Surely might every network be constructed according to a learner's dictation.

How else might an optimal lens be chosen? To be equipped with many alternations beforehand gives clearance over the flow through which to seek for solutions that truly (work). For code swiftly follows design.

The nature of observation and linking lies at the heart of the programs and their designs, and there are many technologies existent for deducing the one from the other. Sending rockets to mars is one way to become familiar with the planets. To deploy submarines is one way to become familiar with planetary depths. Another way is to seek for theory and to observe the plain a bit more sharply, and thereby deduce all the rest. Everything connected with everything is an idea not void, and so does the matter of asking the right questions, and answering every question that one might bring to rise be a mere matter of seeking the right notions and deducing thereafter. This is the idea of preconceptualization. An observer is an interpreter, i.e. conceptualizer, and subsequent preconceptualizer as one completion. For conceptualization does there exist many implementations.

Compatibility One. For compatibility, investigate conventional settings in the current framework. For exhibitability, apply the current framework to settings that extend conventional settings. Comptabitibility setting examples are to be found on canvas, usual book examples on any subject. Exhibitability setting example: visualize in graphs and deploy virtual agent over graph for any sought after deduction.

These written works dictate the direction, code swiftly follows.
(Mathematical software and software follow, software without design is mere computation without aim, mathematics is implied in all written works (see references for details), software is implied in all written works (see references for details), the design that the written works contain lie at the core, any convoluted language bears mountain beneath visible tip, there are constraints to which one must adhere. The

written works dictate the direction, code swiftly follows.)

Hypersymplectic illustration One. Illustration of generalization by considering hypersimplices. Let ordinary state-space be denoted by $\mathcal{S}$. Consider $s' \in \sigma(\mathcal{S})$ and take scaled hypersimplex over s' for $s*$ in new state-space $\mathcal{S}'$. Hypersimplex denotes duratio of device in each ordinary state in collection of states (discretization for realizability). Definition action-space by direction of shrinkage, direction of expansion, fluctuation of hypersimplex (restricted to those with sum bounded by constant L), length of expansion. Definition loss by error in generation. Definition r by reward configuration over losses between states and corresponding action (one simple example is Loss reduction)

Example. First (and quick) technicality check by solving mathematical problems. The following is preliminary and serves to illustrate. There is more to come, deductions more 'advanced'.

Sample. *Let location equal acceleration. Express location in terms of time. This problem can be expressed in Differential Equations. The below is one simple illustration, and more complexity might follow similar lines. Clear example for utility in solution is efficiency gains for software-encoding of movement reinforcement-learning agent.*

One. We have that $x(t) = e^{tA}x_0$ with A diagonalizable, hence $x(t) = C\,\mathrm{diag}(e^{\lambda t})\,C^{-1}x_0 = \sum_{j=1}^{m} v_j \exp(\lambda_j t)c_j$ with $c_j = C^{-1}x_0 e_j$ and $v_j = Ce_j$.

Two. Given $\partial^2 y(t)/\partial dt^2 - y(t) = 0$ and $x(t) = (y(t), \partial y(t)/\partial t)$. Then $\partial^2 y(t)/\partial t^2 - y(t) = \partial^2 y(t)/\partial t^2 + \partial y(t)/\partial t - \partial y(t)/\partial t - y(t) =: A'\partial x(t)/\partial t + B'x(t) = 0$. Extend matrices A' and B' to invertible matrices to obtain extensions A and B respectively for $\partial x/\partial t = A^{-1}Bx(t) = Cx(t)$ for $C := A^{-1}B$. Solution is given by $x(t) = e^{tC}x(0)$. Illustratory extensions $A' \mapsto A$ and $B' \mapsto B$ are given by $A = \left[[1\ 0]^\top\ [-1\ 1]^\top\right]$ and $B = \left[[-1\ 0]^\top\ [1\ 1]^\top\right]$.

Three. Let $A = \left[[\lambda\ 1]^\top\ [0\ \lambda]^\top\right]$. Then $A^n = A^n = \left[[\lambda^n\ (n-1)\lambda^{n-1}]^\top\ [0\ \lambda^n]^\top\right]$. Induction: $A^n A = \left[[\lambda^n\ (n-1)\lambda^{n-1}]^\top\ [0\ \lambda^n]^\top\right]\left[[\lambda\ 1]^\top\ [0\ \lambda]^\top\right] = \left[[\lambda^{n+1}\ n\lambda^n]^\top\ [0\ \lambda^{n+1}]^\top\right] =: A^{n+1}$, indeed.

Four. Solution for $\partial x(t)/\partial t = Ax(t)$ with $x(0) := x_0$ is given by $x(t) = e^{tA}x_0$. Note that $A = \left[[1\ 1]^\top\ [1\ 0]^\top\right] \overset{A}{\mapsto} \left[[2\ 1]^\top\ [1\ 1]^\top\right] \overset{A}{\mapsto} \left[[3\ 2]^\top\ [2\ 1]^\top\right] \overset{A}{\mapsto} \left[[5\ 2]^\top\ [3\ 2]^\top\right] \overset{A}{\mapsto} \left[[8\ 5]^\top\ [5\ 3]^\top\right] \overset{A}{\mapsto} \ldots \overset{A}{\mapsto} \left[[34\ 21]^\top\ [21\ 13]^\top\right] \overset{A}{\mapsto} \ldots \overset{A}{\mapsto} \left[[F_n\ F_{n-1}]^\top\ [F_{n-1}\ F_{n-2}]^\top\right]$. For general form A^k use diagonalizability of A. Denote the 'Golden Ratio', as known from Fibonacci Sequence, by ϕ. Eigenvectors v_1, v_2 of A are $v_1 = [+\phi\ 1]^\top$, $v_1 = [-\phi\ 1]^\top$ with respective eigenvalues $\lambda_1 = \phi$ and $\lambda_2 = \phi$. Next solve $x(t) = c_1 e^{\phi t}[\phi\ 1]^\top + c_2 e^{-\phi t}[-\phi\ 1]^\top$ for c_1, c_2. Fix $t = 0$ to obtain the system of equations $[1\ \ 0]^\top = [1\ 0]^\top = c_1 e^{\phi t}[\phi\ 1]^\top + c_2 e^{-\phi t}[-\phi\ 1]^\top]$. Expression for $c_1, c_2, x(t)$ is given by $c_2 = (e^{-\phi t}\phi - e^{-2\phi t}\phi)^{-1}$, $c_1 = -(e^{-\phi t}\phi - e^{-2\phi t}\phi)^{-1}e^{-2\phi t}$, $x(t) = -(e^{-\phi t}\phi - e^{-2\phi t}\phi)^{-1}e^{-\phi t}[\phi\ 1]^\top + (e^{-\phi t}\phi - e^{-2\phi t}\phi)^{-1}e^{-\phi t}[-\phi\ 1]^\top$.

Matrix-exponentials sample computations. Consider

$$A^k = \begin{bmatrix} \lambda^k & (k-1)\lambda^{k-1} \\ 0 & \lambda^k \end{bmatrix}.$$

Next define,

$$N = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix}.$$

Assume compatibility of matrix dimensions in all computation.

$$\begin{aligned}
\tilde{A} &= \lambda I + N \\
&= \begin{bmatrix} \lambda & 0 & \dots & 0 & 0 \\ 0 & \lambda & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \lambda & 0 \\ 0 & 0 & \dots & 0 & \lambda \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \\
&= \begin{bmatrix} A & 0 & \dots & 0 & 0 \\ 0 & A & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & A & 0 \\ 0 & 0 & \dots & 0 & A \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \\
&\overset{\tilde{A}}{\mapsto} \begin{bmatrix} A^3 & 0 & \dots & 0 & 0 \\ 0 & A^3 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & A^3 & 0 \\ 0 & 0 & \dots & 0 & A^3 \end{bmatrix} + \begin{bmatrix} 0 & \lambda^2 & 0 & \dots & 0 & 0 \\ 0 & 0 & \lambda^2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \lambda^2 & 0 \\ 0 & 0 & 0 & \dots & 0 & \lambda^2 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \\
&\overset{\tilde{A}}{\mapsto} \begin{bmatrix} A^4 & 0 & \dots & 0 & 0 \\ 0 & A^4 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & A^4 & 0 \\ 0 & 0 & \dots & 0 & A^4 \end{bmatrix} + \begin{bmatrix} 0 & \lambda^3 & 0 & \dots & 0 & 0 \\ 0 & 0 & \lambda^3 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \lambda^3 & 0 \\ 0 & 0 & 0 & \dots & 0 & \lambda^3 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \\
&\overset{\tilde{A}}{\mapsto} \dots \overset{\tilde{A}}{\mapsto} \begin{bmatrix} A^n & 0 & \dots & 0 & 0 \\ 0 & A^n & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & A^n & 0 \\ 0 & 0 & \dots & 0 & A^n \end{bmatrix} + \begin{bmatrix} 0 & \lambda^{n-1} & 0 & \dots & 0 & 0 \\ 0 & 0 & \lambda^{n-1} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \lambda^{n-1} & 0 \\ 0 & 0 & 0 & \dots & 0 & \lambda^{n-1} \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix}.
\end{aligned}$$

Recall matrix exponential e^{tA} given by

$$e^{tA} = \begin{bmatrix} e^{\lambda} & \left(\frac{\lambda-1}{\lambda}\right) e^{\lambda} \\ 0 & e^{\lambda} \end{bmatrix}.$$

Matrix exponential $e^{t\tilde{A}}$ is given by

$$e^{t\tilde{A}} = \begin{bmatrix} e^{\lambda} & 0 & \dots & 0 & 0 \\ 0 & e^{\lambda} & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & e^{\lambda} & 0 \\ 0 & 0 & \dots & 0 & e^{\lambda} \end{bmatrix} + \begin{bmatrix} 0 & \left(\frac{\lambda-1}{\lambda}\right) e^{\lambda} & 0 & \dots & 0 & 0 \\ 0 & 0 & \left(\frac{\lambda-1}{\lambda}\right) e^{\lambda} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \left(\frac{\lambda-1}{\lambda}\right) e^{\lambda} & 0 \\ 0 & 0 & 0 & \dots & 0 & \left(\frac{\lambda-1}{\lambda}\right) e^{\lambda} \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix}.$$

First Practical Detail. Mathematics, SoftwareEngineering, ComputerScience. Simulation Engines: Game Engines: Unity, UnrealEngine, UbiSoftFramework, Anvil. Engines for simulation with pre-installed setting: Celestia, Unity, NVIDIA PhysX, Ansys Fluent, Advanced Simulation Library (OpenSource). Computers: Cloud: LambdaLabs, etc. Scripts deployable in open-world games or engines of simulatio, mainly for capture of virtual setting interconstruction: Heaviest PC Game-type: RTX-9040 Orion Predator Acer, Vector GPU Desktop/Workstation. Gadgets: IoT STM3224 Discovery kit node, ESP32-Azure IoT Kit, IoT-aided artifacts, oculary-autorotators, engine simulation before robotics, oculary must be very high definition.

Mathematics, SoftwareEngineering, ComputerScience. Simulation Engines: Game Engines: Unity, UnrealEngine, UbiSoftFramework, Anvil. Engines for simulation with pre-installed setting: Celestia, Unity, NVIDIA PhysX, Ansys Fluent, Advanced Simulation Library (OpenSource). Computers: Cloud: LambdaLabs, etc. Scripts deployable in open-world games or engines of simulatio, mainly for capture of virtual setting interconstruction: Heaviest PC Game-type: RTX-9040 Orion Predator Acer, Vector GPU Desktop/Workstation. Gadgets: IoT STM3224 Discovery kit node, ESP32-Azure IoT Kit, IoT-aided artifacts, oculary-autorotators, engine simulation before robotics, ocular must be very sharp.

First Architectural Design Detail: First Architecture 1: Legend over hyperparameters. [Architectural Design Detail: Architecture 1: Legend over hyperparameters.] First expression of project in terms of popular notions.

- **[CrossDL] = [CrossCNNRNN]**
- **[WalkerDRL]** requires care in definition space S, framework POMDP possible.
- Test architectures by deployment, i.e. grade constructions by simulation
- ($\mathcal{D}$-dependent) obs. $\mathrm{CNN}_{\mathcal{D}_f}$ (single convolutional layer)
- **n** (Iteration count application convolutional layers)
- **r'** (Consideration trade-off training multiple small networks versus training one big network)
- λ_n (Consideration trade-off training multiple small networks versus training one big network)
- s (Under optionality of hyperparameter r', subscript $\mathcal{D}$ takes various definitions). In other words, different hyperparameter inclusion for Vision, different

state representation. $[CrossDL]^{s}_{\mathcal{D}_f}$ might be 1 network, with $\mathcal{D}$-dependent obs. $\text{CNN}_{\mathcal{D}_f}$ stacked after the previous, resulting in an optimal network that might bear the intepretation of an optimal lens)

- h_s (definition state history, either static past observations over time intervals, or past sequences of static observation within one time-frame, corresponds to definition s)

- **Directions** Learner achieves performance objectives deducible from heuristic visual. New objectives and devices following from old construction of old devices.

First Architectural Design Detail: VisionCNN Architecture, Architecture 2: Legend over hyperparameters. [Architectural Design Detail: VisionCNN Architecture, Architecture 2: Legend over hyperparameters.] First expression of project in terms of popular notions.

- **CNNoptimization** (hyperparameter in [Cross- $\text{DL}^{s}_{\mathcal{D}_f}$] for optimization, or training, optimal CNN for s. Examples Defintion Loss measure, hyperparameters in optimization algorithm (velocity, parameters).)

- **RNNoptimization** (hyperparameters in [Cross- $\text{DL}^{s}_{\mathcal{D}_f}$] for optimization optimal CNN-RNN for s. Examples Definition Loss-measure, hyperparameters in optimization algorithm. Definition reards, for example by specification of loss-configuration between current state and subsequent state, one example specification is the difference in Loss)

- **a** (type of movement for search preconceptualization, depends on search-space, and device of application. See BipedalWalker-v2, e.g. application by drone similar, or traversal dataspace, settlement LunarLander)

- $\mathbf{p}(\mathbf{a}|\mathbf{s})$ (deterministic mainly, stochasticity due to uncertainty in environment or device of application)

- $\pi_\theta(\mathbf{a}|\mathbf{s})$ **optimizable network**. (Hyperparameters in Learning algorithm for optimization $\pi_\theta(a|s)$, immediate exploration dictates on-policy learning, for example by Proximal Policy Optimization (PPO))

- **Directions** Learner achieves performance objectives deducible from heuristic visual. New objectives and devices following from old construction of old devices.

First Speculative Avenues of Pursuit I. Speculative Avenues of Pursuit I. First expression of project in terms of popular notions.

- Multiple observers, multi-agent RL

- Observer sophistication and device complexity

- Multi-agent, multi-observer swarm deployment

- Output-type numeric encoding for alignment with convention

- Continuous observer interpreter readjustment
- Deployment in settings of sure expected functionality
- Generalization to new settings to be ancored in deployment to setting of sure expected functionality
- Input sensor interpreter other than convolutional neural network
- Intantaneous setting transformation upon retrieval
- Multi-agent, multi-observer for both input and output with static central control coordinator
- Every technological vehicle, under specialization to its setting can function as an observer, either autonomous or not.

First Speculative Avenues of Pursuit II, One: Network Design Initialization. First expression of project in terms of popular notions.

- Multiple observers, multi-agent RL over setting of deployment and simulated mirages for amplification of subsets of setting features
- GlobeMap, MolStruct, Computational chip can all serve as settings, or layers in simulated mirage of greater setting from where these can be extended
- Results in deep generative network will serve as a basis for observing beyond the plain
- Observation encoding/decoding is not necessary for functionality, but is necessary for generation correctness
- In every multi-agent deployment can a central coordinator steer over the above points
- Simple settings will serve as foundations for complex setting for correctness verification
- Human interface through, e.g. linguistic expression of output

First Readily Constructible Avenues of Pursuit One. Readily Constructible Avenues of Pursuit. First expression of project in terms of popular notions.

- Learners over virtual space, including IoT Networks
- All the notes written up to the present
- Evaluation performance implicit in (D)RL and DL losses. Heuristic visual serve as heuristic guidance.
- Specification of objectives
- Device programming and assembly ready for deployment
- Monitoring Learner activity and performance

- Deployment in virtual setting Unity PhysX on small computer
- Deployment in open-world games on big computer
- Deployment over IoT with or without external gadget
- Multi-agent extension towards first most general implementation
- Deployment on every usual task in data analysis
- Continuous lens-re-adjustment over visor movements
- Addition or subtraction of complexity in Learner's construction, more powerful device can give and run more complex learner in the same framework current, whether this is good is to be observed from performance monitoring
- Gain in sum of existing trade-offs by lowering movement capacity for higher observation capacity
- Lens construction is setting-dependent and setting-distances are readily defined, or learned
- Written documentation for technical specification in lens construction by sequential placement of elements
- User-defined travel for observation beyond the plain
- Acquire necessary devices

First Stages of development. First full practical implementation in starting simulation software can occrue according to the steps below. Unity is one starting engine, and the below outlines implementation concretualization.

1. **Script Writing.** Any notebook would suffice for Script Writing. We settle for the most rudimentary digital notebook that **Apple** has to offer.
2. Manually crafted virtual space deployed machine in assembly CNN-RNN-DMARL or CNN-RNN-DRL, using any text-editor or code-editor
3. (Engine) prepared virtual space deployed machine in assembly CNN-RNN-DRL with oculary-agent in assembly CNN-RNN-DRL with (Engine) oculary-agent role taken by built-in oculars by installing code-editor agent with (Egine) launcher. Evolve from simple environments to complex environments, e.g. $2D \rightarrow 3D$. Account for the possibility of speculative improvements, implementable, e.g., in game engine.
4. Explore different Egines, beyond MuJoCo's Unity-plug-in or the PhysX-Unity subEngine for deployment virtual open-world simulation of complex and detailed environments for great information capture, and much space for potential low loss processing, i.e. potential comovement. Improve machines if improvement found by abnormality in Engine capacity or monitored results or new inspiration.

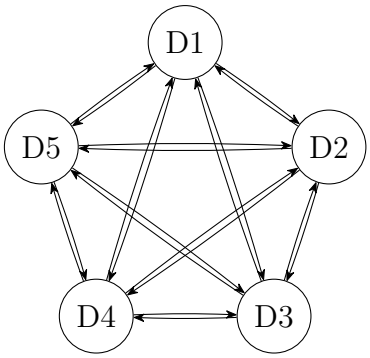


Figure 1: Complete bidirectional graph for a so-called DL-StarSystem of the fifth order, vertices represent intralinks, edges represent interlinks. One obvious generalization lies in DL-StarSystems of the n'th order, representable by a nomplete bidirectional graph with n vertices. See file `Ytractorfunctional.py` in repository `DLStarSystem`.

5. Explore machine-deployment in the real-world, programmed according to machine-deployment in the virtual-world.

Star-systematic configuration for illustration of direction for extension in Deep Learning. (Creation: Construction-Cycle 1) Having access to all that surely knows currently, infer the rest as good as possible. Features, settings, assumed. Microscale $\rightarrow$ Exascale. Datatypes: Numeric Array, Image, Sound (convertible to Image), Numeric Array $\rightarrow$ Image by Image Generator, DeepDream. Build then extend, learning links of links, learning links, notions: order of links; links between environments. The links in every intralink-vertex are the datasets, and every intralink-vertex in generated by their respective aspect of deployment in Data-space. Start rudimentarily with 5 nodes. Implement and report performance. Example. Inquiry towards $D5$, access to $D1$ and $D2$: try $D1 \rightarrow D5$, $D2 \rightarrow D5$, try $D1 \rightarrow D2 \rightarrow D5$ and $D2 \rightarrow D1 \rightarrow D5$. The complete bidirectional graph in Figure 1 is labeled by $(D_1, D_2, \ldots, D_5)$, and each of the two edges between any two distinct nodes is connected by edges in both directions hither and thither respectively.

Inquiring into $D2$ requires some knowledge on $D2$ to be extended from depoyment in data-space. Call the above, Starlink Order 5. Starlink Orders 1, ... 5 work. Generalization to higher orders would but require a heavier computer. Inquire into feature detection. Among the datasets are the vertex-intralinks. The above graph is bidirectional in every edge. See https://github.com/Ali-Al-Habsyi/DLStarSystem for full implementations. We have experimental evidence, soon to be reported, of superior performance. Herefrom follow extensions that one might indeed deem valuable. *The extension DL-StarSystem serves to illustrate approach towards an extension that fits the preceding descriptions. It bears* ***some*** *basis in mathematics, and one might engineer the system's specification of deployment over dataspace towards grand utility of the system, as one would engineer any ordinary neural network. To bring higher mathematics is our first pursuit. Advanced machines in software and hardware follow. The development of these machines are our next pursuit.*

In keywords, Directions Forward Include. *(full stops "." serve as semicolons ";")* Continuum supremacy (analogous to 'Quantum Supremacy'). Continuum computing. In the plain can new realms be discovered by considering continuity (Ricci Flow). Representation with ∞-information density, as $\mathcal{D}_f$ has ∞-information den-

sity. Computation must have throughput-level ∞. Simulation Science captures Computer Science and Discrete Mathematics in the conventional sense. Continuum Science captures all non-discrete mathematics and emphasizes the notion of continuity and ∞. Consider interface, or portal, between simulation in full continuity, as captured by reality (or universe), or simulation in partial discretization, as captured by simulation science, to transition from one to the other. Biological, neuro-organic, computation ensures capacity to continuum compute, as evolution happens on platform reality, or universe. For practical demonstration apply portal for transition from continuum to dicrete realm. For research in paradigm of continuum computation as benchmarked by conventional paradigms of computing, apply portal for transition from discrete realm to continuum. For true discovery on universe in only stationary work-station, consideration of continuity is essential. Consider exemplary Millennium Prize Problem Poincaré Conjecture. Important in 'Important Accomplishments' in mathematics is relative. Many open problems are more straight-forward than they seem. Millennium Prize Problems are worthy of hunting down.

In keywords, Quick Notes. *(full stops "." serve as semicolons ";")* Information density $\mathcal{D}_f$ is without bound. "Mathematics of the continuum": Analysis on Manifolds, Interface/Portal Realms Discrete/Continuous, Modern Algebra, Modern learning (toward Continuum learning), Modern Quantum Computing and Discrete Mathematics (towards Continuum Computing and Continuum Observing), (Analysis on Manifolds captures Multivariate Analysis), (Modern Algebra captures Linear Algebra), (Interface Realms Discrete/Continuous captures Calculus and Integration Theory), (Essential to the interface Realms Discrete/Continuous is the notion of continuity, most variants of which are slight variations of the definition: f is continuous at a if $[\forall \epsilon > 0 : \exists \delta > 0 : |x - a| < \delta \implies |f(x) - f(a)| < \epsilon]$). We seek to generalize. How might such a generalization read? In continuum observing do we seek to extract data $\mathcal{D}_f$, infinitely progressive in level of detail from where data-extraction occurs. Great addition to the level of complexity in existing notion of continuity hint towards useful notions for '$\mathcal{D}_f$ as a continuum'. Introductory Mathematics of the Continuum. Complex Analysis: formulation traversal Mandelbrot set, Analysis on Manifolds: Greene's Theorem. Algebra of differentials d. Assume first observal with infinite capacity and deduce therefrom. (Oculary) Restrictive parametrics in e.g., Extent of scope, Definition of Data-capture, (point-wise, or, e.g., Manifold-encoding) (Statistics over Machinal Configurations) Representations of $\mathcal{D}_f$, Information throughput. Continuum Computing, Continuum Learning, Assume perfect machine machine, the ideal scenario. All capacity for information storage and implementation of restrictive parameters. Required are algorithms of the continuum. One route of construction Mathematical Machine. Define Mathematical Machine in mathematical terms, and deduce all further. A practically implementable version of mathematical machine is a mere variant of mathematical machine. Supposition. Introductory Continuum Computing. Pre-collection of mathematical machines over the continuum, just as quantum computing is theory of "Quantum Algorithms". Introductory Continuum Learning. Big milepoint is in formulation of Algorithms of the continuum. This gives Mathematical Ocular, by assembly of continuum observer, continuum computer. Research. Algorithms of the continuum, Extraction of data from the continuum, Formulation Mathematical Machine, Continuum Learning: Learning over the continuum, The statistics of the continuum. The job is now

to seek the right notions. E.g., Line → Manifolds, Stochastic processes → Manifold Stochastic Fluctuation processes for initialization modeling $\mathcal{D}_f$.

In keywords, Mathematical Analysis. *(full stops "." serve as semicolons ";")* Complex Analysis: formulation traversal Mandelbrot set, Analysis on Manifolds: Greene's Theorem. Algebra of differentials d. Assume first observal with infinite capacity and deduce therefrom. (Oculary) Restrictive parametrics in e.g., Extent of scope, Definition of Data-capture, (point-wise, or, e.g., Manifold-encoding) (Statistics over Machinal Configurations) Representations of $\mathcal{D}_f$, Information throughput.

PDE's, Analysis on Manifolds, Interface/Portal Realms Discrete/Continuous, Modern Algebra, Modern learning (toward Continuum learning), Modern Quantum Computing and Discrete Mathematics (towards Continuum Computing and Continuum Observing), (Analysis on Manifolds captures Multivariate Analysis), (Modern Algebra captures Linear Algebra), (Interface Realms Discrete/Continuous captures Calculus and Integration Theory), (Essential to the interface Realms Discrete/Continuous is the notion of continuity)

Research. Algorithms of the continuum, Extraction of data from the continuum, Formulation Mathematical Machine, Continuum Learning: Learning over the continuum, The statistics of the continuum. The job is now to seek the right notions. E.g., Line → Manifolds, Stochastic processes → Manifold Stochastic Fluctuation processes for initialization modeling $\mathcal{D}_f$

Relevance towards the above interoperable. Contribution magnitude measurable by inclusion problem as Millenium Prize Problem. Modelling, Analysis on manifolds. Stochasticity: Stochastic Processes and Stochastic Analysis. Solve open problems that are slightly related to the above. Analysis on manifolds, Ricci flow with surgery on three-manifolds, the entropy formula for the Ricci flow and its geometric applications. Stochasticity: Stochastic Processes and Stochastic Analysis.

In consequence, formulate statements that relate to $\mathcal{D}_f$ as a continuum. Soon shall the notion of 'continuum' be made precise. Follow pattern exemplary in development QFT. On the way, solve problem Yang–Mills existence and mass gap. Analysis on manifolds, Stochasticity: stochastic processes and stochastic analysis. Solve open problems that are slightly related to the above. ClayMathInstitute. Either prove or disprove.

Analysis on manifolds. Ricci flow with surgery on three-manifolds, The entropy formula for the Ricci flow and its geometric applications → Poincaré conjecture stochasticity: stochastic processes and stochastic analysis, Complex analysis, Riemann Hypothesis. This, in fact, is not at all difficult.

Essay Millenium Problems towards solutions Milenium Problems towards solution of this Essay, this File. *This document contains the essay on the final problem to be solved.*

- Differential Equations → Navier-Stokes Equation,
- Complex analysis → Riemann Hypothesis,
- Quantum Field Theory → Yang–Mills existence and mass gap (Development QFT (Quantum Field Theory) $\iff$ Development '$\mathcal{D}_f$ as a continuum'),
- Discrete Mathematics: Computing and Algorithms → P versus NP,

- Algebra's of differentials (see Analysis on manifolds) → Hodge conjecture, Birch and Swinnerton-Dyer conjecture,
- Analysis on manifolds → Specification definition '$\mathcal{D}_f$ as a continuum'.
- Stochastics → Specification definition 'Statistics of the continuum $\mathcal{D}_f$'.

CROSS-DL$^s_{\mathcal{D}_f}$ Implementation One

[CrossDL $\tilde{:=}$ CrossCNNRNN]$^s_{\mathcal{D}_f}$ is one example for a form that $[CrossDL]^s_{D_f}$ can assume. Consider a [DLStarSystem]$^s_{\mathcal{D}_f}$ for which we have experimental evidence of functionality in utilization of [CrossDL]$^s_{\mathcal{D}_f}$ in crossCNNRNN where to utilize [CrossCNNRNNStarSystem]$^s_{\mathcal{D}_f}$ and extract states in manner such as optimal online exploration, for a DLStarSystem with optimal states for inferring the D_1 of StarSystem [CrossCNNRNNStarSystem]$^s_{\mathcal{D}_\{}$. Losses are defined by quality of inferring D_1 for $D_2, D_3, D_4, \ldots$ and an infinite-state DRL-agent might provide the optimal policythrough which to D_1, by providing generalizations of DGN through [CrossCNNRNNStarSystemRNNDRL]$_{\mathcal{D}_f}$. This is highly implementable, now lready, "readily implementable absolute minimum". Use Keras for a starting software-library.

The experimental evidence of [DLStarSystem]$^s_{\mathcal{D}_f}$ must not be meddled with; herein lays one implementation of [CrossDL]$^s_{\mathcal{D}_f}$. This is how we implement our first construction in code-construction-2. The results of [DLStarSystem]$^s_{\mathcal{D}_f}$ are worthy of publication and extension towards the grandest scales. An implementation of DLstarSystem exsists already, thus extend trhe found implementation by generalization and maximal-scaling. Note that $\mathcal{D}_f$ is arcane, and so is [DLStarSystem]$_{\mathcal{D}_f}$. The depth of an environment must be learned. Thus start with Optimal Online Ecploration towards optimal nodes $\{D_i\}_{i\in\mathbb{N}_+}$ to integrate in [CrossCNNRNNStarSystem]$_{\mathcal{D}_f}$, and deploy an overarching DRL-agent that traverses over the StarSystem by hopping oer $\{D_i\}_{i\in\mathbb{N}_+}$. Why Deep Learning? Because it tweakably works. Why [DLStarSystem]? Because it tweakably works. Meddle not with the supreme results of your experimental endeavours.

Thus, generalize DLStarSystem towards its utmost extent of discrete generalization $\{D_i\}_{i\in\mathbb{N}_+}$. Interlink-rder, Intralink-order, etc, no MAGIC-NUMBERS, but hyperparameters in new DL-Starlinks, Let $D_1 := D_T$ be the target Node, representable by any absorber of Vision, Vision might give representation of all the order data-nodes $D_2, D_3, \ldots$ i.e. $\{D_i\}_{i\in\mathbb{N}_+-\{1\}}$, and deploy one DRL-agent that for the optimal configuration of $\{D_i\}_{i\in\mathbb{N}_+-\{1\}}$-CNNRNNDRL deployed over $\mathcal{D}_f$ for the optimal inference and correction for problems in momentary machine learning. The software library Keras sufficies to implement the aboce. How might such an implementation occur? Vision: $V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9$, Data: $D_1, D_2, D_3, D_4, D_5, D_6, D_7, D_8, D_9$, $D_1 \tilde{:=} D_T$ (target-node) $\{D_i\}_{i\in\mathbb{N}_+-\{1\}} = \{D_i\}_{i\in\{2,3,4,\ldots,9\}} \cup \emptyset$ foudn through optimal online exploration with losses $L_i := \text{Loss}(D_T, D_i)$, $i \in \{2, \ldots 9\}$. This is the step wherein to search for data-environment-nodes. after which to collect all in [CNNRNNStarSystem]$^s_{\mathcal{D}_f}$, after which to deploy [CNNRNNStarSystemRNNDRL]$^s_{\mathcal{D}_\{}$. So, to summarize, OOE → [CNNRNNStarSystem]$^s_{\mathcal{D}_f}$ → [CNNRNNStarSystemRNNDRL]$^s_{\mathcal{D}_f}$. This is standard implementation of [CrossDL]$^s_{\mathcal{D}_f} \rightleftharpoons$ [WalkerDRL]$^s_{\mathcal{D}_f}$.

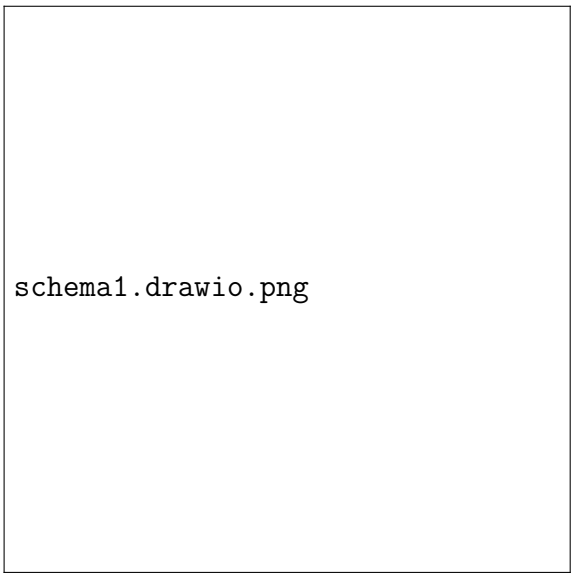
Imagine one telescope-satellite equipped with OOE to search according to the path of a Brownian particle. Environment-nodes $\{D_i\}$ can have an uncountable basis

such as $\mathbb{R}$, and this is where BipedalWalker gains its applicability and relevance. Thus $\mathcal{D} := \{D_i\}_{i\in\mathbb{R}^2}$, the implementable optimal nodes might define one pattern in $\mathbb{R}^2$, signifying a map of relevant environments to infer through $\mathcal{D}_T$. This gives $\mathcal{D}_{OOE} = \{\mathcal{D}_i\}_{i\in\mathbb{R}^2\cap M}$ and applying $\mathcal{D}_{OOE} = \{\mathcal{D}_i\}_{i\in\mathbb{R}^2\cap M}$ gives one ClusterSystem of uncountable many nodes that themselves must bear intralinks that are to be cross-interlinked for $[\text{DLClusterSystem}]^s_{\mathcal{D}_f}$. Now define $\mathcal{S}_i := \mathcal{D}_{OOE}$ for a DRL-agent that travels over $[\text{DLClusterSystem}]^s_{\mathcal{D}_f}$ and there optimally infers D_T. The basis of of this is that DLStarSystem woth by ecperimental evidence, reports for which are to follow. The DLClusterSystem is a remarable avenue of pusuit. The attainment of Data mgiht however require slight work. It somehow works. It somehow works. There is talk on two-stage deployment of [CrossDL] $\rightleftharpoons$ [WalkerDRL]. How would this occur? $[\text{CrossDL}]_{\mathcal{D}_f}$ from first encounter of $\mathcal{D}_f$, plain observation. Mine for data-entanglements through some Optimal in [Walker-DRL]. This is an important research pursuit.

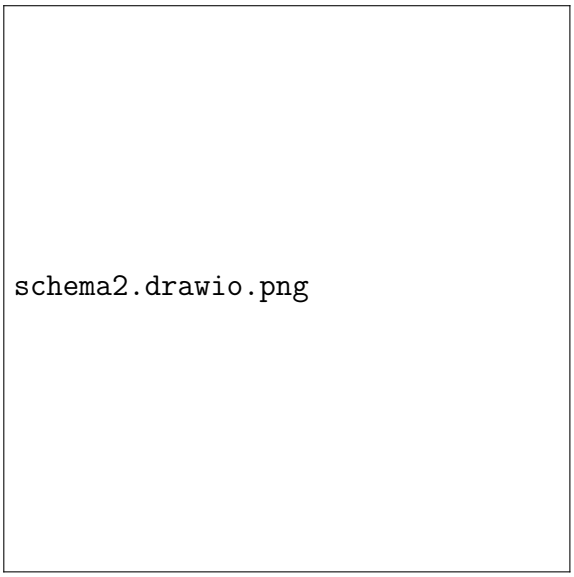
Let Algorithmic Set-up be defined. Encredit Google for their Alpha-series. Consider the following

1. CNNAssembly by ChipPlacementDRL
2. E.G. AlphaFoldAssembly through AlphaChip
3. DRL: OptimalOnlineTraversal
4. HyperparametricConsideration, this is Automizable

The later enumeration is almost trivial. We focus on the DLClusterSystem. This is to give the required software-architecture for code-construction cycle 2. One idea is to first set-up the DL-StarSystem and let a DRL-agent travel over this DL-StarSystem. The DL-StarSystem can also be a DL-ClusterSystem. Permutative Integration gives various avenues of inference of D_T, so vary avenue of inference by traversal over permutative integration configurations. This would give a hypersymplectic agent, and multiple obserers would not bear improvement here. An additional RL-agent might expand the ClusterSystem by searching for data-nodes that give the optimal implementation in the resulting DRL equipped DLClusterSystem. Schematically this reads as



Note that there is no limit to CompletionExpansionDRL and that the above implementation serves to bear hints towards real implementation. For example



Et cetera. This is implementable software. Implementable software is the point if this section. A latex-coded diagram would indeed be better. This follows soon.

DYSON-SPHERIC THEORETICAL ARCHITECTURE

What other models do there exist for **CROSS-DL**$^{s}_{\mathcal{D}_f}$? Let us bring exploration, highly software-architectural (i.e. mathematical). There are many implementations for optimal online exploration, to the end of generating new data-nodes over which to traverse. OOE can read as follows. RL-Configuror with state-space (s^3, a^3, loss-configuration($\{L\}$), restant hyperparameters) and action-space bringing change in configuration of [NacigatorDRL] reading as $(s, a, \text{Loss-configuration}(L))$. Exploring starts over $\{s, a, \text{Loss-configuration}(L)\}$ by EnvironmentLearning gives configuration $\{s, a, r\}$. NavigationLearning over $\{s, a, r\}$ gives $\{\pi\}$. The generation of new DataEnvironments happens through TargetGeneration by following the momentary specification of π. Now repeat from the start by again EnvironmentLearning and continue until some treshold criterion or guaranteed eventual convergence, although convergence might take a long time to reach. The point hereof is to give a specification of a two-layered optimal-online-explorer OOE. Note that this is deployment 1.3 in outline's given specifications. Implementation in software follows readily.

In the preceding section have we mentioned HypersimplecticDRL, but we have not specified it. Let us attempt to bring a first specification. Let the 'ordinary state space' be denoted by $\mathcal{S}$. Consider $s' \in \sigma(\mathcal{S})$ and take the scaled hypersimplex over s' for s^* in the new state space $\mathcal{S}'$. Hypersimplex denotes duration of the device in each oprdinary state in a collection of states. (Discretization for realizability). The action-space $\mathcal{A}$ can be defined by (1) direction of shrinkage, (2) direction of expansion and (3) fluctuation of the hypersimplex (restricted to those hypersimplices bounded by constant L, length of expansion. The rewards $\mathcal{R}$ are given by reward configurations over losses between states and corresponding action (one simple example is Loss reduction), and the loss is defined by the error in generation (additively enforcable over $\mathcal{S}'$)

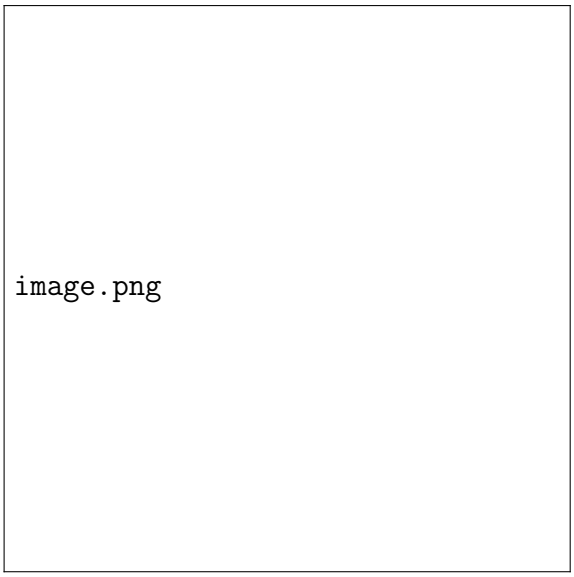


Figure 2: This is a graph with 64 vertices. Each D is a data-environment. We hereby represent learning learning learning learning learning learning learning. There are 31 datanodes in total. The extent of interlinks is determined by the number of datanodes.

We must emphasize that we do MachineLearning and that RL and DL are main paradigms. We create new paradigms, and solve reality. The reference of CompletionExpansionDRL serves to configure and optimize both DLClusterSystem and HypersimplecticDRL, this componenty might be the OOE. Note that higher order CompletionExpansionDRL components are OOE components as well. My mistake was to not share my architectural progress and so be relocated to the hospital, with overleaf shall this mistake not be made again. We seek for models of $\mathcal{D}_f$ through [CrossDL]. Consider the diagram

insert diagram here

This represents learning learning learning learning learning learning learning, i.e. learning up to its seventh order to stem from a DLStarSystem with 31 nodes. So [CrossDL] works for $\mathcal{D}_f$. What else might work for $\mathcal{D}_f$? Stochastic Manifold Fluctuation Processes with manifolds of a very high dimension. For this have we books. Thus let us endeavour books. We seek to ultra-enhance 'the scope and scale of conciousness', and this is not an endeavour that is baseless, there is truth to it. All the subdivisions of Creation are concerned with Mill-construction, satellite-deployment, rocketry, Data-collection, Mobile-smartphone application omni-presence. Creation overarches it all. An AI company, and the biggest of all companies. We start through Overleaf and GitHub. Little poetry, much hardcore software-architecture and mathematics. I would call myself an engineer more than a mathematician, and my mathematics serve the end of constructing the omni-observant general super artificial intelligence. The author is aware of the arrogance that he might convey through his tone, and reclaims none of it. So, hardcore software-architecture and mathematics in Overleaf, construction through GitHub, NASA and Creation Inc. later. First cycle: 10 papers, 1 emergent machine. Second cycle: 100 papers, 2 emergent machines, etc. Literature sift we go. The final machine must be the

Omni-observant Omni-communal Omni-informative Artificial General Superintelligence. Reality must be encoded through some mathematics structure, constinuous matrices are highly informative. **There is to come a mighty scientific blast soon. A mighty mathematical blast. This will give 10 scripts of circa 50 pages each. We construct our first machine of 3 million lines of code. We hope for this blast to occur before newyears. We will need much paper, such onto HEMA first. JLMR, JHEP main scientific journals. DeepMind main corporate AI. NASA main space agency. Lo and behold, dear Earth. This will be glorious. Everything for the omni-observant Artificial General Super Intelligence.** *We apologize for the delay.* Yes, we do apologize for the delay. But, in due time shall the OAGSI be completed, and have I attained victory. **In due time shall the OAGSI be completed, first need I to sift through all of JMLR. This might take days.**

$$\times \to + \to \otimes := [\times \times \times \times \times \times \times \times \times \times] \to \oplus \to \bigotimes \to \bigoplus \to \dots \tag{0.1}$$

We proceed as follows (for aye). Recall that we do Physics and AI.

1. Read all of journals JMLR and JHEP
2. Initiate Creation Cycle One as dictated under SIMULATIONII; SIMULATIONII and ProjectProjectEssay Creation-code-designCycleII is the exercise, JMLR is the literature.
3. Find quality literature and reiterate Step 2 with new literature and new cycle of Creation

So read much first; we are now at step One. Reading all of JLMR (https://www.jmlr.org/) carefully does take some time; we request some patience from Earth. (+DEEPMIND/NASA for the practical ,DL,DRL,RL,LTP,ESL,ECTR, MODERN-PHYSICS / CM / QM / GUAGE-THEORY / QFT / ETC.)

We are aware of the need to to keep the public up-to-date, and we shall make every effort to complete reading JMLR and the supplementary ML material. We subsequently proceed in a fashion similar to DeepMind way of labour. The software-architectural scripts that are to follow soon shall be highly mathematical. Hence our sift through JMLR. Alright, here we go again. And finished with JMLR we are. Now onto the build-up of Creation in Theory. We bring DLClusterSystem to full completion. What is CREATION? Let us bring substance to our ideas. He is worth the protection, ladies and gentlemen. **Dear reader, I need a few days off due to testicular recovery. After the slight pause and after full recovery, shall I be back at this again with might and main.** And recovered we have, software-architectures are to follow soon, new software is to follow soon. Alright, here we go :). WE ARE NOW AT CONSTRUCTION PHASE 2.