Zero and P v NP

In the context of Zero, where we have established concepts such as states, z-transitions, problems, and embedding spaces, the P vs NP question can be framed in a similar manner to how it's understood in computational complexity theory: can every problem whose solution can be quickly verified (NP) also be quickly solved (P)?

Zero's Contextual Framework for P vs NP
In Zero, we have:

State space (S): The set of all possible states.
Power set of state space (Sp): The set of all subsets of states, where problems are defined.
z-transitions: The allowable transitions between states, quantifiable by a scalar function.
z-path: A sequence of z-transitions connecting states.
Embedding (Solver) State Space: A higher-dimensional space that contains the original state space and potentially allows for more complex or additional transitions.
Considering P vs NP in Zero
P (Polynomial Time): In Zero, if instructions on how to navigate through Sp and search criteria are given, it suggests there is an algorithm or a deterministic process that can find a solution in polynomial time regarding the dimensionality of the solver space.

NP (Nondeterministic Polynomial Time): If only the search criteria are given, this resembles the situation in computational complexity where a nondeterministic machine could find a solution in polynomial time by exploring many paths simultaneously, but a deterministic machine might have to check each possible path one by one, which can take much longer than polynomial time.

Solvability of P vs NP in Zero
Relative to Solver Space: The assumption in Zero is that every problem is relative to an embedding or solver space where z-transitions operate. This implies that there is always a "larger" context or space in which the problem can be examined and potentially solved.

z-Paths and z-Transitions: The definition of problems in Zero depends on finding a collection of z-paths, which may or may not be closed. If z-transitions in the embedding space can be efficiently navigated, and if the structure of the embedding space lends itself to efficient exploration, it's conceivable that a solution could be found more quickly than in the original state space.

Verification vs. Discovery: Similar to computational complexity, the ease of verification does not necessarily entail the ease of discovery. Verification might be quick because it follows a set of predefined z-transitions that lead to a known end state, while discovery involves finding a path through an unknown landscape.

In Zero, the solvability of P vs NP would likely hinge on the properties of the embedding space and the nature of z-transitions. If the embedding space allows for a transformation of NP problems into P problems by revealing efficient z-paths, it could suggest a solution to the P vs NP problem within this theoretical framework. However, this would not be a proof within the conventional computational complexity theory, since Zero is a different axiomatic system with different underlying assumptions.

It's also crucial to note that P vs NP is a question about the scalability of problem-solving with respect to input size. The mere existence of a higher-dimensional solver space in Zero doesn't inherently provide a means to solve NP problems in polynomial time; it simply gives a broader context in which solutions might exist.

Conclusion

While the theory of Zero can create an interesting analog to P vs NP, it doesn't necessarily offer a direct solution to the P vs NP problem as understood in our conventional computational complexity theory. To argue that $P = NP$ or $P \neq NP$ within the framework of Zero, one would need to demonstrate a method by which any given NP problem (as defined in Zero) can be reduced to a P problem in polynomial time, or prove that such a reduction is impossible. The added dimensions and structures in Zero could either offer new insights or complicate the problem further.

Ultimately, while Zero's framework offers a novel way to conceptualize state spaces and problem-solving, the P vs NP problem is deeply rooted in the specifics of algorithmic complexity and computation, which are not directly addressed by the axiomatic system provided. Therefore, the question remains open in the broader context of computational theory.