K Means

Clustering

Full Technical

Documentation

Cyrus365.com

0. Contact & Support

🎁Marketplace: https://www.unrealengine.com/marketplace/en-US/product/k-means-clustering

💬Discord: https://discord.gg/xE2qkuwjwH

📧Email: ue_support@cyrus365.com

5 Mins Quick Start Guide:

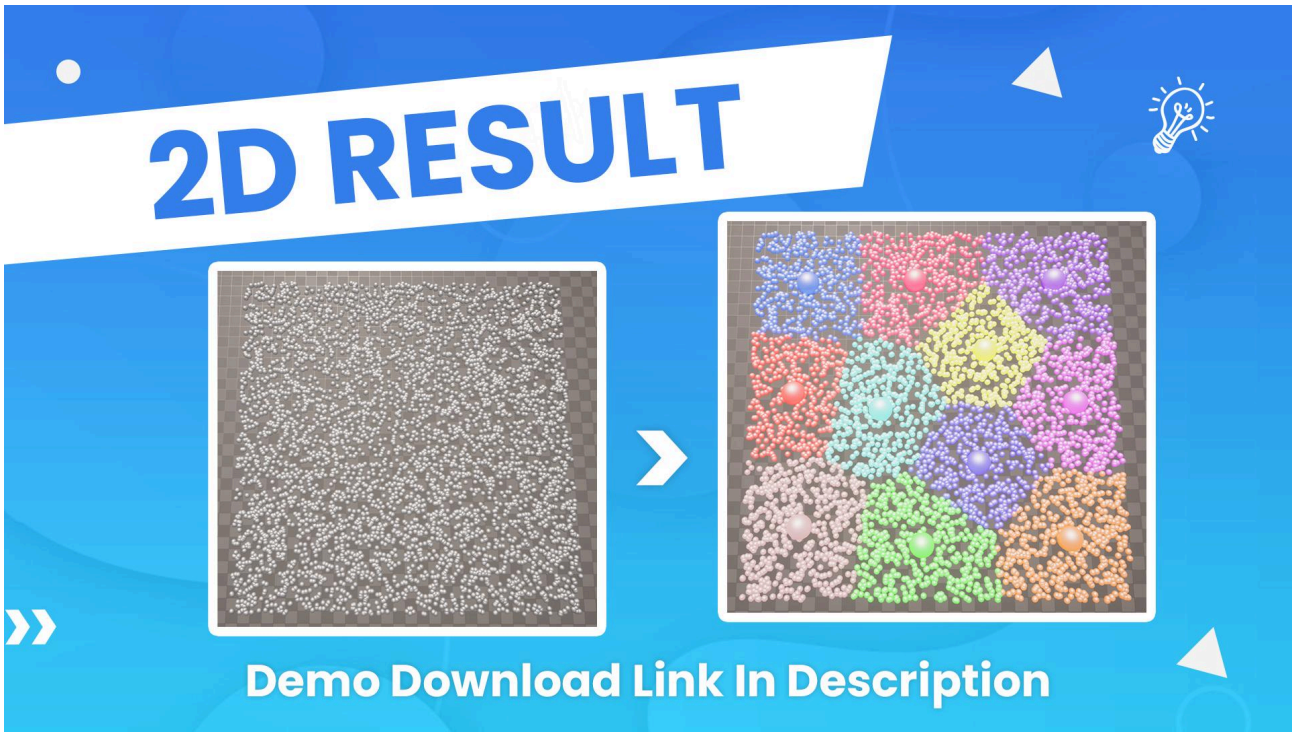
K Means Clustering - 5 Mins Quick Start Guide

---

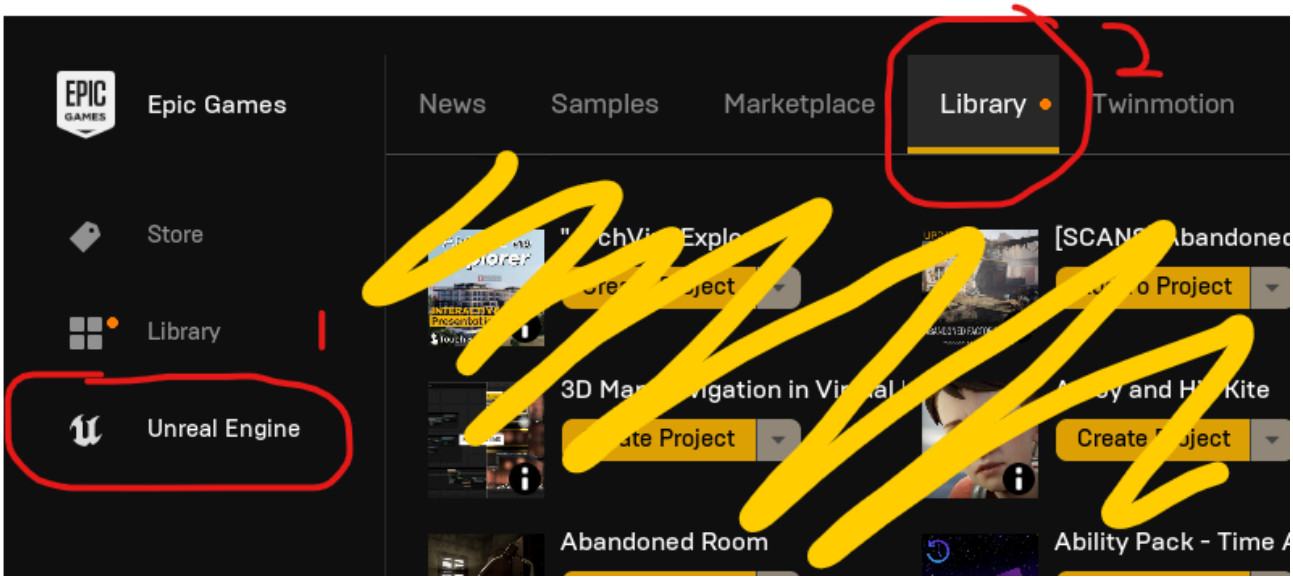
1. Plugins Usage Example

This documentation primarily focuses on the 3D implementation of the K Means Clustering plugin. The functionality for both 2D and 3D remains the same, with the only difference being the inclusion of (Z) coordinates in the 3D variant.

Please note that while the examples and explanations emphasize the 3D aspects, the principles and usage are consistent across both 2D and 3D implementations.



2. Installation

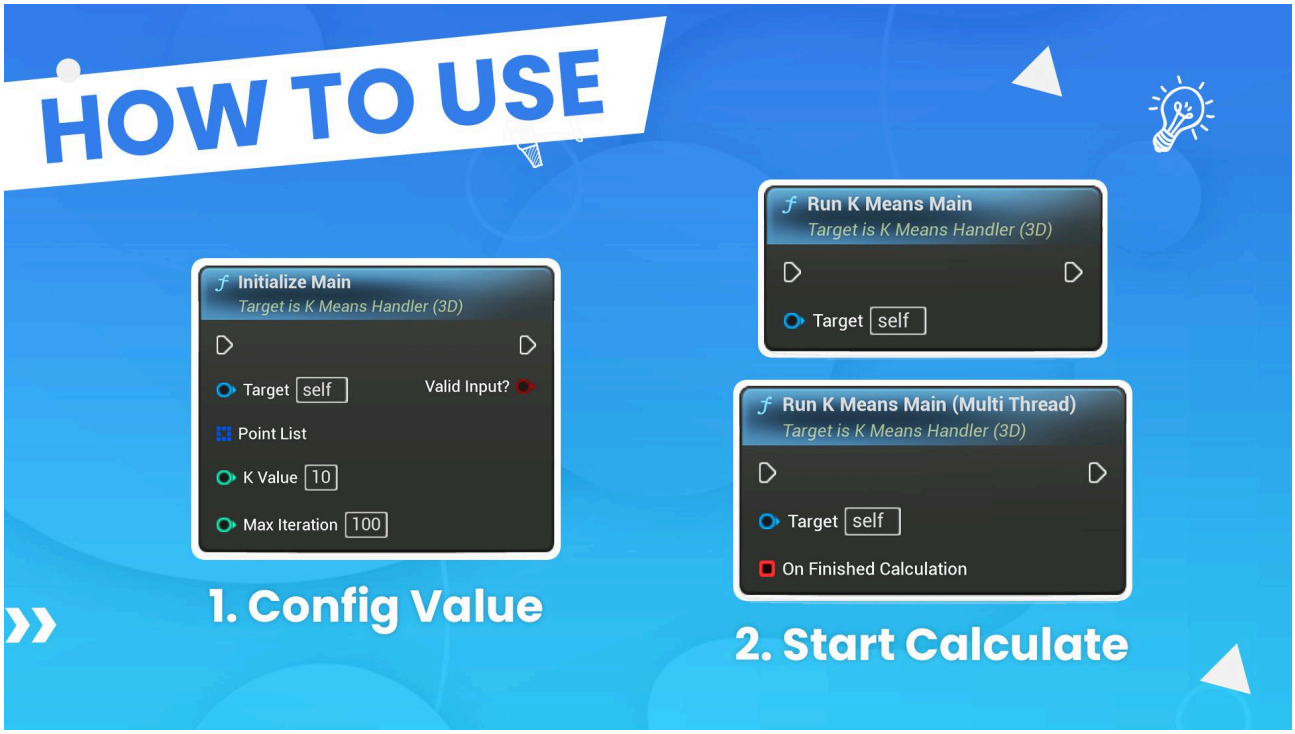


1) Open Epic Games, Click “Unreal Engine”
2) Click “Library”
3) Find “K Means Clustering” in vault and click “Install”

After opened UE project, enable Plugin Before Use

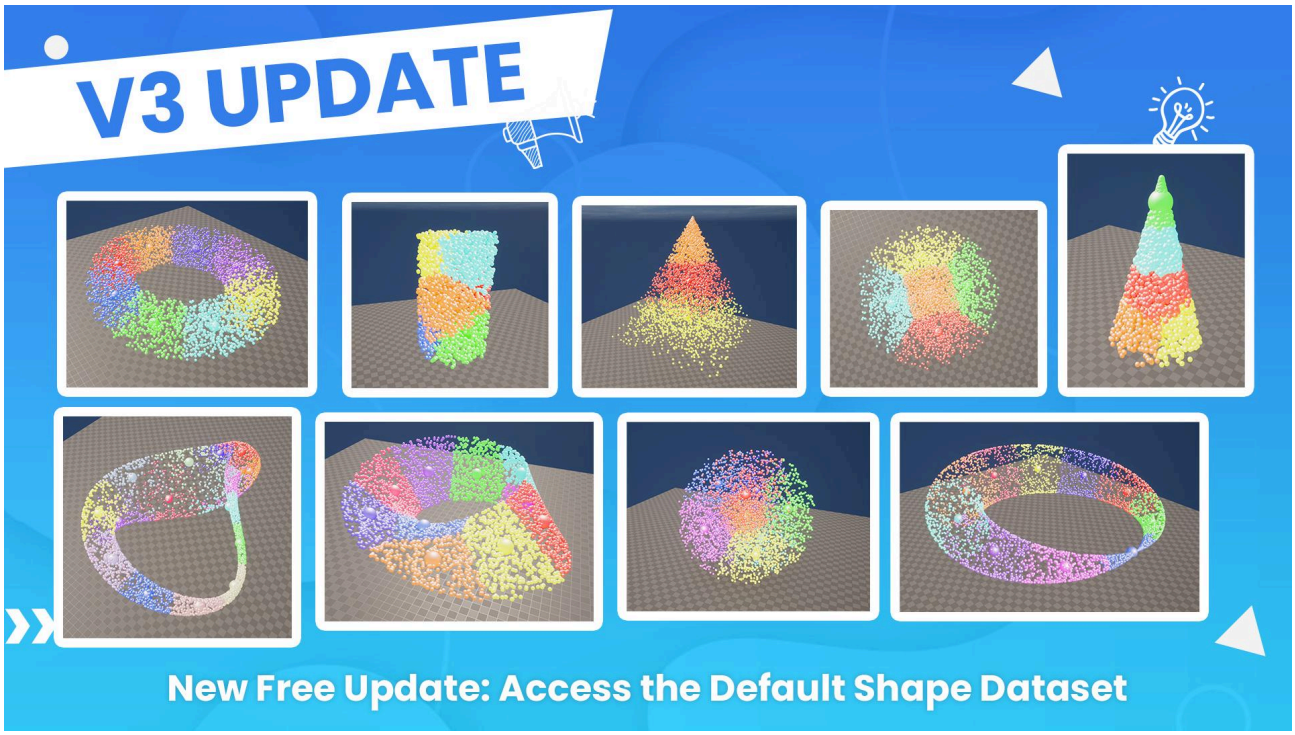
Go to “Edit” -> “Plugins” -> Search for “K Means Clustering” and check it.

3. Main BP Node



Keywords to Search In BP: "KMC"

Version 3.0 Update – Access the Default Shape Dataset



2D Shapes:

1. General (RandomPoints2D)

2. Circle (RandomPointsInCircle2D)

3. Normal Distribution Circle Shape

(RandomPointsInNormalDistributionCircleShape2D)

3D Shapes:

1. General (RandomPoints3D)

2. Sphere (RandomPointsInSphere3D) (Shell Included:

RandomPointsOnSphericalShell3D)

3. Pyramid (RandomPointsInPyramid3D) (Shell Included:

RandomPointsOnPyramidShell3D)

4. Torus (RandomPointsInTorus3D) (Shell Included: RandomPointsOnTorusShell3D)

5. Cylinder (RandomPointsInCylinder3D) (Shell Included:

RandomPointsOnCylinderShell3D)

6. Cone (RandomPointsInCone3D) (Shell Included: RandomPointsOnConeShell3D)

7. Conical Frustum (RandomPointsInConicalFrustum3D) (Shell Included:

RandomPointsOnConicalFrustumShell3D)

8. Klein Bottle - 3D Pinched Torus

(RandomPointsOnKleinBottleShell3D_3DPinchedTorus)

9. Klein Bottle - Bottle Shape (RandomPointsOnKleinBottleShell3D_BottleShape)

10. Möbius Strip (RandomPointsInMobiusStrip3D)

For Beginners: What is Blueprint?

1. Right click on any blueprint graph that you want to implement.
2. Search the “Blueprint Name” (Blue color field) and click on it.

   Or Search **"KMC"** (the short-term keyword of this plugin), it will show all available functions.
3. Please fill the input pin. Some are optional if it has the default value.
   It will show a tooltip guide message when you highlight the pin (Not all input has tooltip, but I am sure that the difficult one will have tooltip).
4. If you got any problem, please go to Discord Support Group for more information:
5. I will be very happy if you rate the product on Unreal Marketplace!

---

4. Step by Step Quick Start Guide

Reference here: K Means Clustering - Quick Start Guide

---

5. Key Class, Method and Property Reference

This plugin supports both 2D & 3D K Means. They have the same functionality, 2D have (X,Y) but 3D have (X,Y,Z). And this document focuses on 3D.

AC3_KMC_Handler3D

Actor that handles running 3D K Means calculations. Add to scene and configure.

- `InitializeMain()` - Primary setup method. Pass points and parameters here.
- `OnCalculationFinished` - Event triggered when calculation completes.
- `RunKMeansMultiThread()` - Start calculation in background thread.
- `GetGroupMemberNum()` - Helper to get count for a specific cluster.

FC3_KMC_Point3D

Struct representing a 3D point with ID and cluster assignment.

- `Id` - String ID that can uniquely identify each point.
- `X/Y/Z` - Numeric coordinates of the point.
- `GroupId` - Cluster this point belongs to.

FC3_KMC_Result3D

Struct encapsulating complete clustering result data. Passed via event.

- Points - Input point list.
- Centers_All - Positions for all cluster centers.
- Centers_Single - Centers containing only one point.
- Centers_Group - Centers containing multiple points.
- MemberList - Mapping of clusters to member points.

Point

A point refers to a single data item that will be clustered. This could represent an AI character location, environment asset position, player movement coordinate, or any other game data you want to analyze.

Internally, each point is an FC3_KMC_Point3D struct containing:

- Id: Unique string identifier you can assign
- X/Y/Z: Numeric coordinate location
- GroupId: The assigned cluster after running K Means

So a point conceptually represents a single positional data item that will be grouped into clusters by proximity.

Center

A center refers to the calculated central position representing a cluster. The number of clusters is defined by the K parameter.

The plugin determines the optimal center positions by iteratively updating them based on which points are closest. Centers end up in the "middle" of clusters.

You can access the final positions via the `Centers_All` list in the result struct. Single point clusters go into `Centers_Single` as a convenience.

Member List

The member list is a mapping from cluster indexes/IDs over to the individual points assigned to that cluster. Essentially, it associates each center with the child points closest to it after convergence.

This is provided in the `MemberList` property of the result. It is useful for iterating through each cluster's members for visualization or further analysis after completing the K Means clustering.

6. Source Code Reference

Class Name	K Means Handler (3D)
C++ Class Name	AC3_KMC_Handler3D
Class Description	Actor that handles running 3D K Means clustering calculations.

Class Name	K Means Handler (2D)
C++ Class Name	AC3_KMC_Handler2D
Class Description	Actor that handles running 2D K Means clustering calculations.

Member Name	Type	Description

LocalDevSettings	UC3_KMC_DeveloperSettings*	Local developer settings instance.
Points	TArray<FC3_KMC_Point[2D/3D]>	Store the list of points for K Means clustering.
Centers_All	TArray<FC3_KMC_Point[2D/3D]>	Store all centers regardless of the number of points they contain.
Centers_Single	TArray<FC3_KMC_Point[2D/3D]>	Store centers that contain only 1 point.
Centers_Group	TArray<FC3_KMC_Point[2D/3D]>	Store centers that contain more than 1 point.
MemberList	TMap<int, FC3_KMC_GroupMember[2D/3D]>	Map representing GroupId to its members.
MaxIteration	int	Maximum times to iterate the K Means algorithm.
KNum	int	K number: How many centers to show in the world.
PointNumber	int	The number of points. Auto-calculated.

Worker	FC3_KMC_Worker[2D/3D]*	Worker object that handles running the K Means algorithm in the background.
bIsInitialized	bool	Flag indicating if the object has been initialized with valid data.
OnCalculationFinished	FC3_KMC_Result[2D/3D]Signature	Callback executed when the background calculation finishes.

Method Name	Type	Description
InitializeMain()	bool	Main method to initialize the K Means clustering object.
InitPoints()	void	Initialize the list of [2D/3D] points.
InitPoints_Random()	void	Initialize points with random data.
InitK()	void	Initialize the value of K (number of centers).

InitCenters()	void	Initialize the starting locations of the centers.
InitCenters_Random()	void	Initialize the starting locations of the centers.
GetRandomCenters()	TArray<FC3_KMC_Point[2D/3D]>	Select random starting positions for the K cluster centers.
InitIteration()	void	Initialize the maximum iteration count.
NearestCenter()	int	Get distance between a point and its nearest center.
Cluster()	void	Assign each point to the nearest center based on their current positions.
Center()	void	Recalculate the positions of the centers based on the points assigned to them.
GroupCenters()	void	Separate centers into single vs. multi-point groups.

CalculateGroupMemberList()	void	Calculate the group member list, mapping GroupId to its members.
IsReadyToRun()	bool	Is the K Means ready to calculation?
RunKMeans()	void	Start to calculate K Means in background.
RunKMeansMultiThread()	void	Start the K Means clustering in a background thread.
IsPointsValid()	bool	Check if the list of points is valid.
GetPoints()	const TArray<FC3_KMC_Point[2D/3D]>&	Get the list of points.
IsCentersValid()	bool	Check if the list of centers is valid.
GetCenters()	const TArray<FC3_KMC_Point[2D/3D]>&	Get the list of centers.

IsCentersSingleValid()	bool	True if Centers_Single is not empty, false otherwise.
GetCentersSingle()	const TArray<FC3_KMC_Point[2D/3D]>&	Get the centers that only contain 1 point.
IsCentersGroupValid()	bool	True if Centers_Group is not empty, false otherwise.
GetCentersGroup()	const TArray<FC3_KMC_Point[2D/3D]>&	Get the centers that only contain multiple(more than 1) point.
GetGroupMemberList()	const TMap<int, FC3_KMC_GroupMember[2D/3D]>&	GroupId
GetGroupMemberNum()	int32	Get the number of members for the given group ID.