Range Quantization

What is Range Quantization

Quantization is the process of mapping values from a large set to a smaller set. This section is concerned with quantization with respect to integer numbers. For example, an integer has 32 bits of data and consequently it is capable of representing 2^32 distinct values. In **many cases we only use a small range** of these distinct values. If we know this small range of values, we may **apply quantization and reduce our information bandwidth**.

> https://www.mathworks.com/help/comm/ug/quantization.html (https://www.mathworks.com/help/comm/ug/quantization.html)

ℹ NOTE

Range Quantization **does not reduce precision while still reducing bandwidth**.

Assume that we have a player in a game world and we know that this player can only move within the XYZ coordinate space of -1024 to +1023. With quantization we can restrict the range of integer values for our player and represent the players position with 33 (3*11) instead of 96 (3*32) bits.

ℹ NOTE

The quantized player position requires 11 bits **per coordinate axis** since we only need to represent 2048 (2^11) distinct values as opposed to 4 294 967 296 (2^32).

How to use Range Quantization

Building on the previous example of a player in a game world it is conceivable that we may define a position vector for our player:

```
public struct PositionVector
{
    int x;
    int y;
    int z;
}
```

We know that for each coordinate axis the player can not move outside of the range -1024 to 1023. It would then be **wasteful to serialize the position vector as follows** (we do not use the full range):

```
// We first define a buffer to seraialize into
PacketBuffer buffer(1024);

// We need a writer to help us with writing into the buffer
PacketWriter writer(buffer);

// Assuming that we have a position vector that we want to serialize
PositionVector position;

// We use the writer to pack each component of the vector
writer.PackInt(position.x);
writer.PackInt(position.y);
writer.PackInt(position.z);
```

Instead, it is advisable to **quantize the position vector with knowledge of the restricted range**:

```
// We first define a buffer to seraialize into
PacketBuffer buffer(1024);

// We need a writer to help us with writing into the buffer
PacketWriter writer(buffer);

// Assuming that we have a position vector that we want to serialize
PositionVector position;

// We use the writer to pack each component of the vector
writer.PackInt(position.x, -1024, 1023);
writer.PackInt(position.y, -1024, 1023);
writer.PackInt(position.z, -1024, 1023);
```

ℹ NOTE

It is required to **use the same range when unpacking as when packing**.