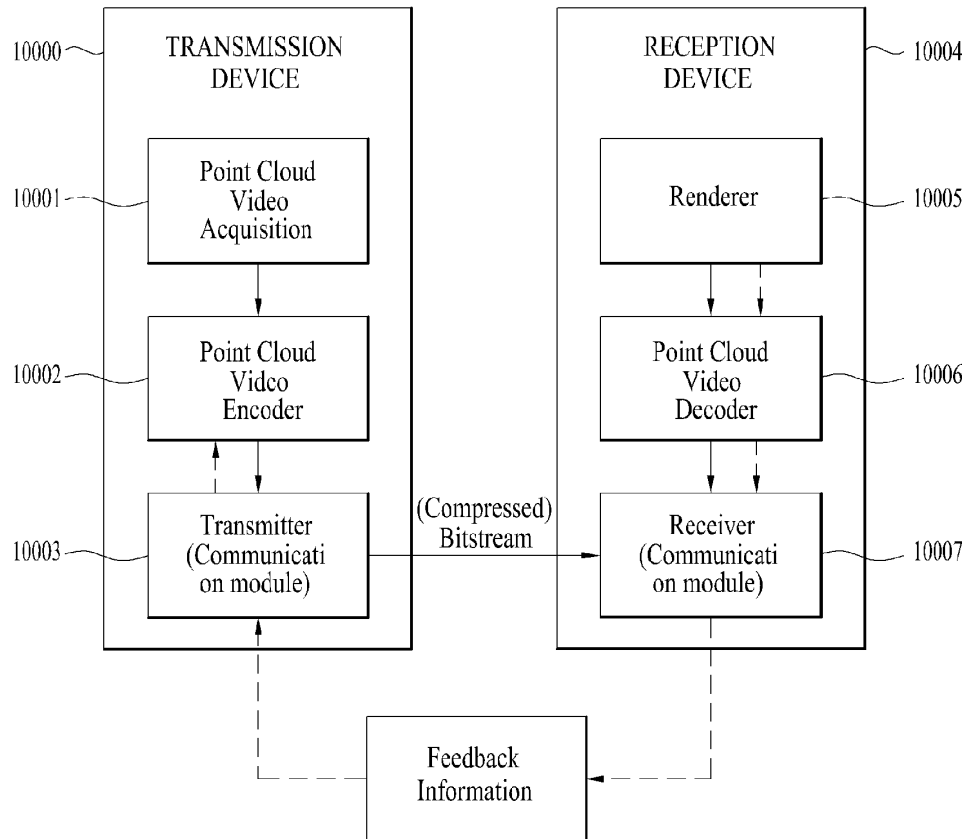
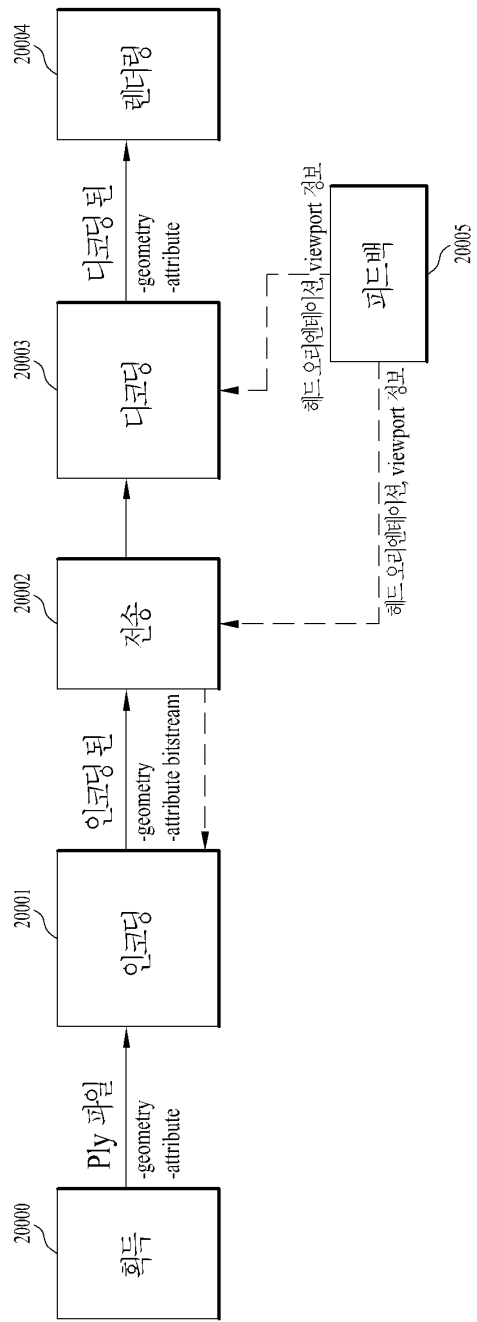


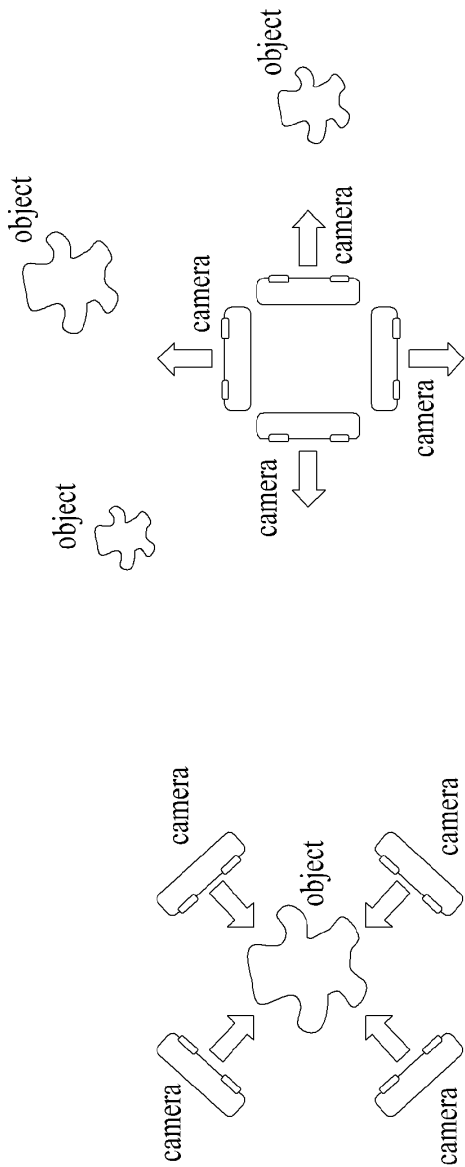
[도 1]



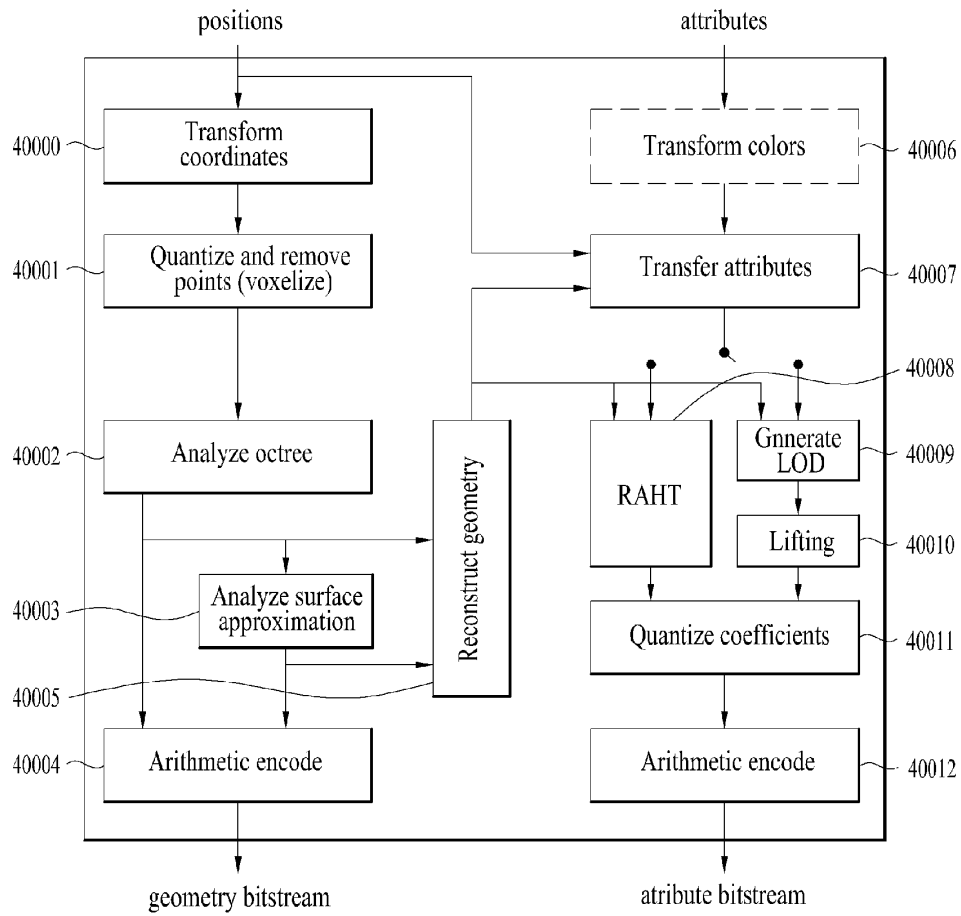
[도2]



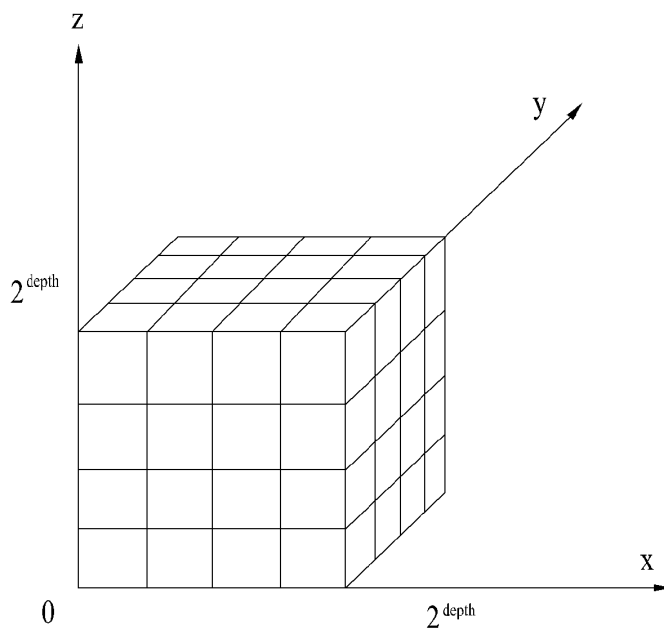
[도3]



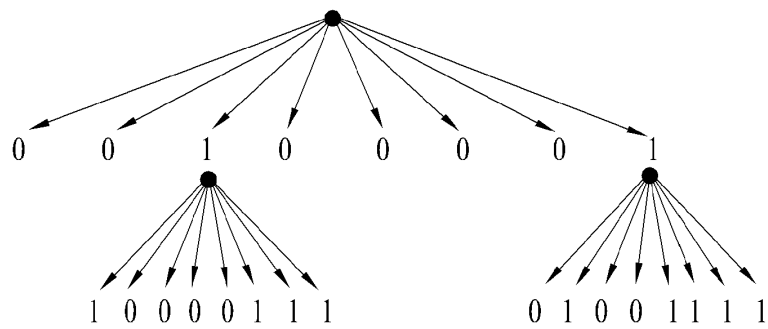
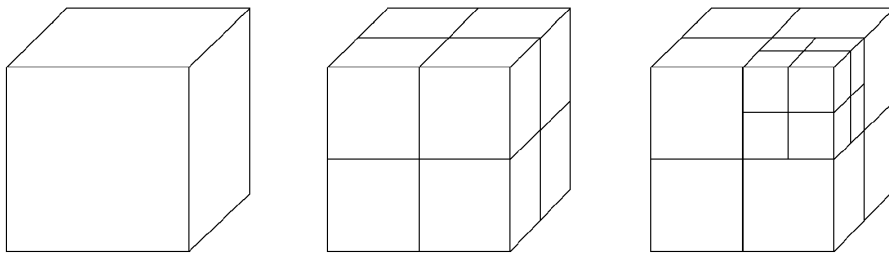
[도4]



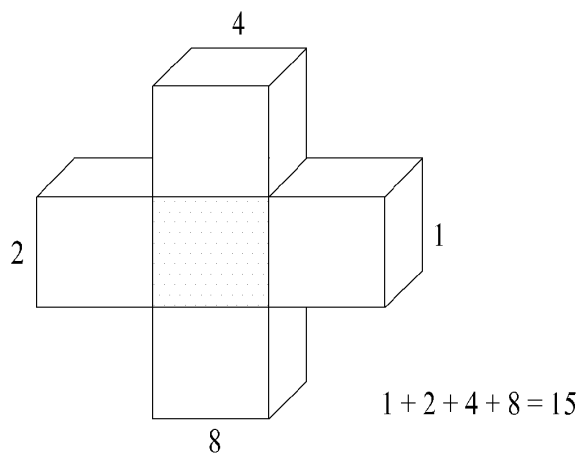
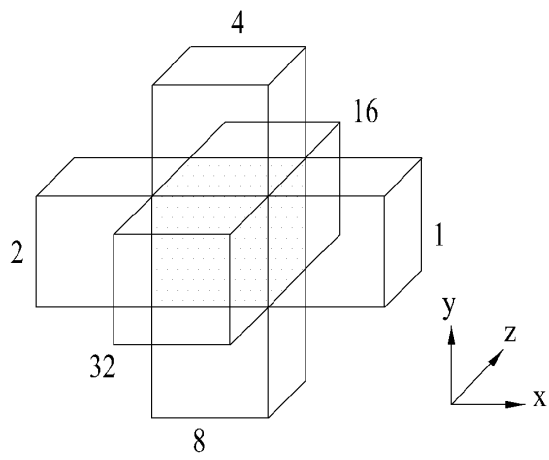
[도5]



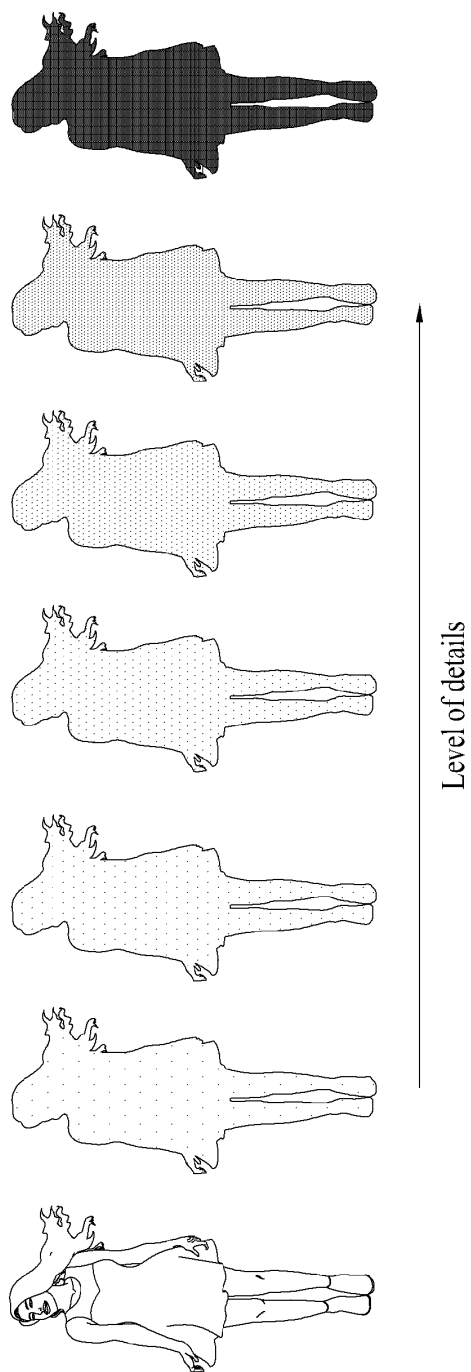
[도6]



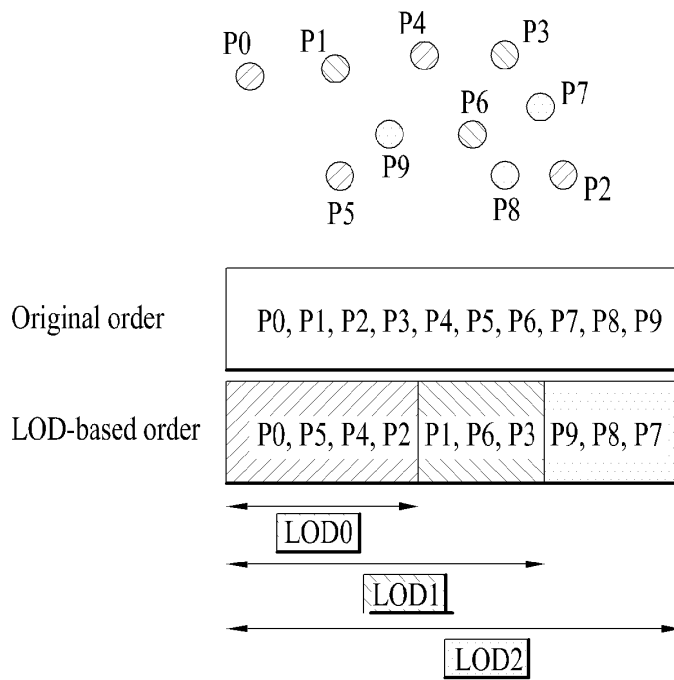
[도7]



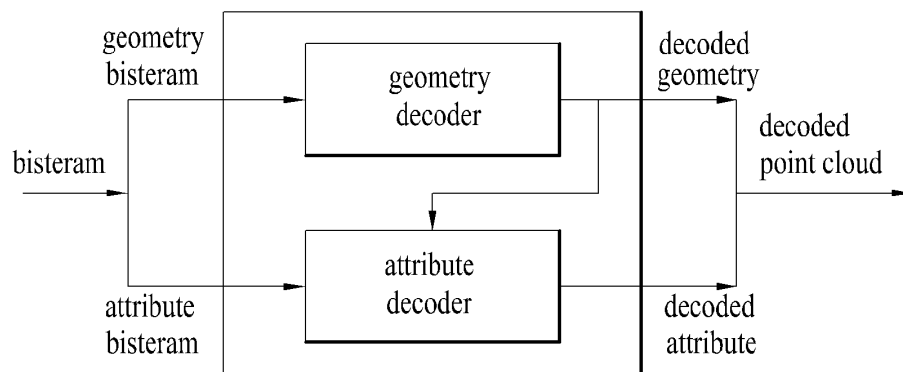
[도8]



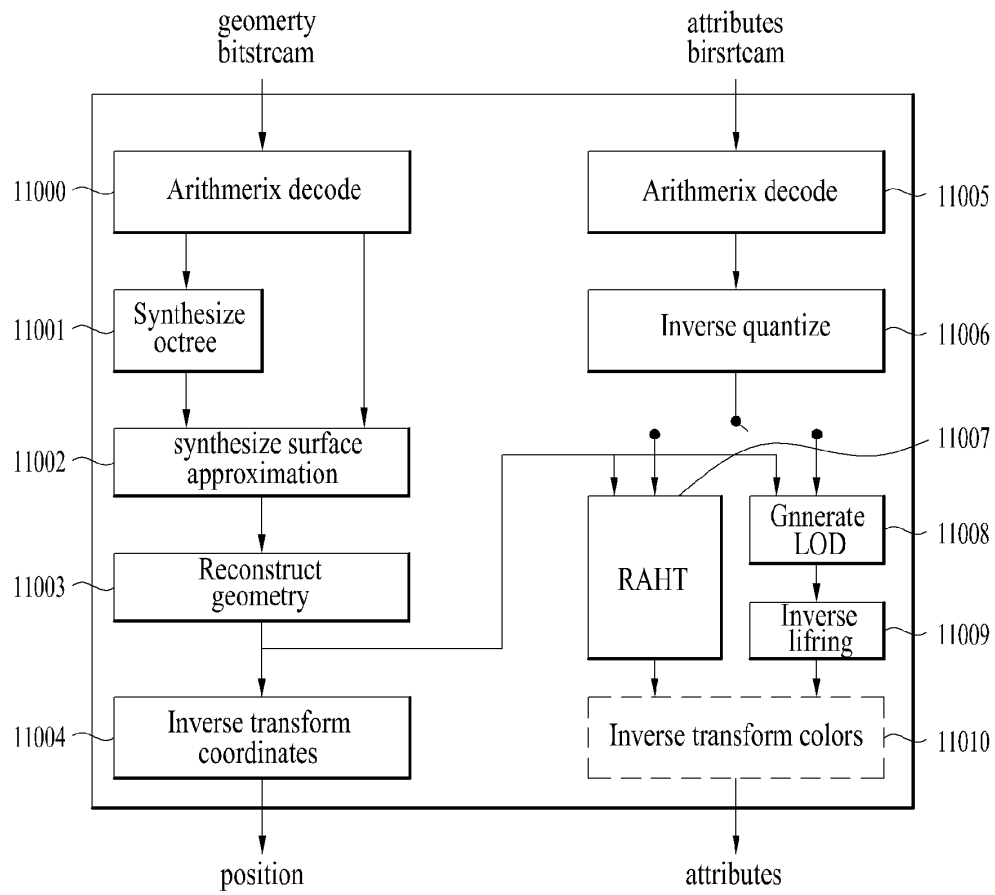
[도9]



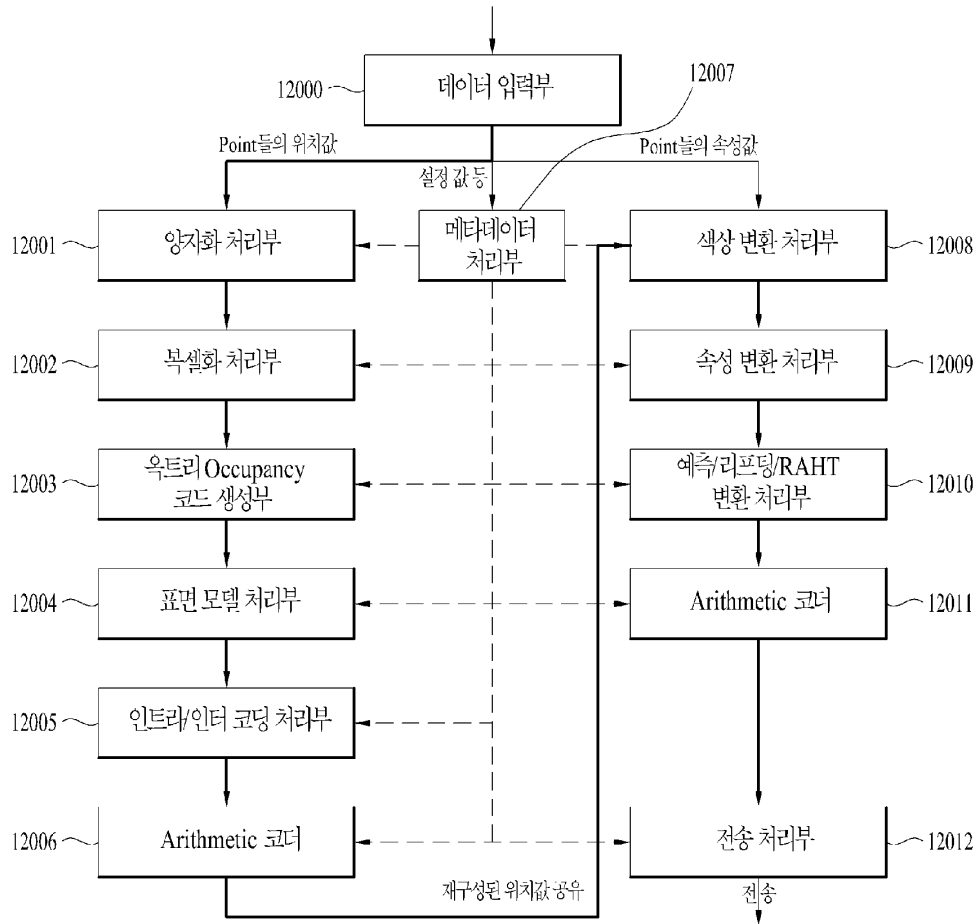
[도10]



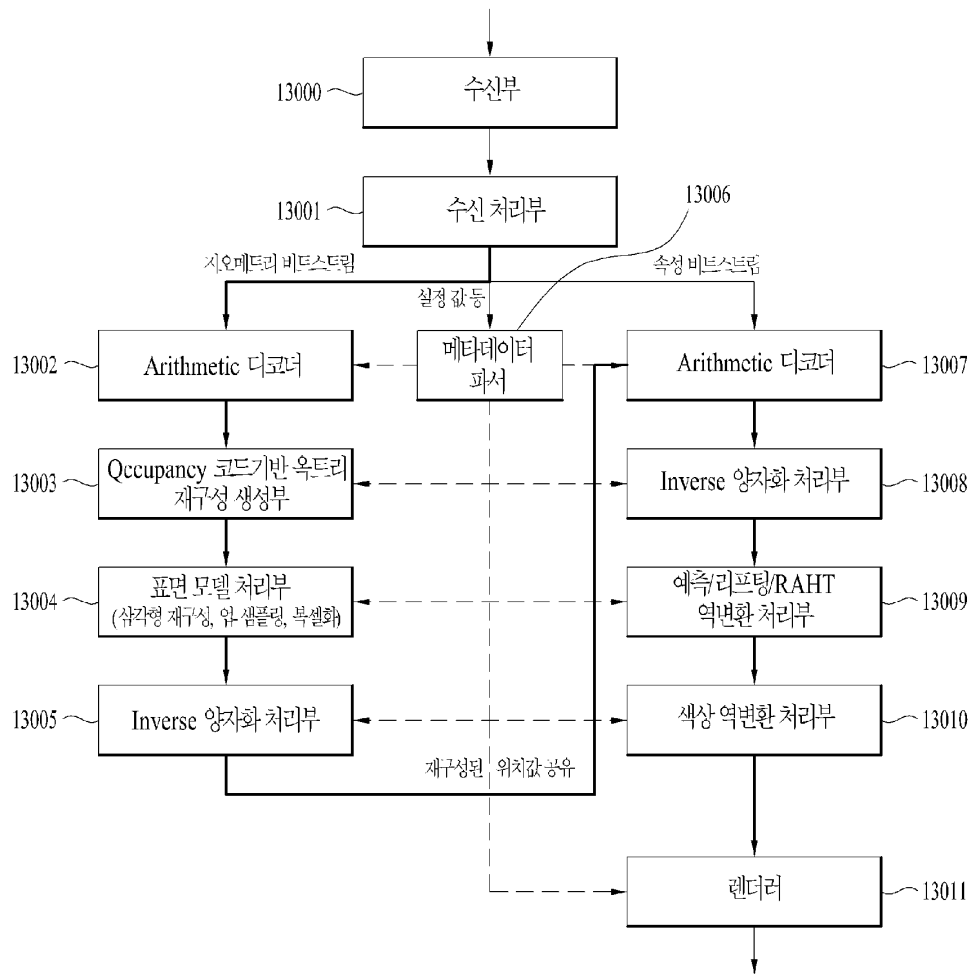
[도 11]



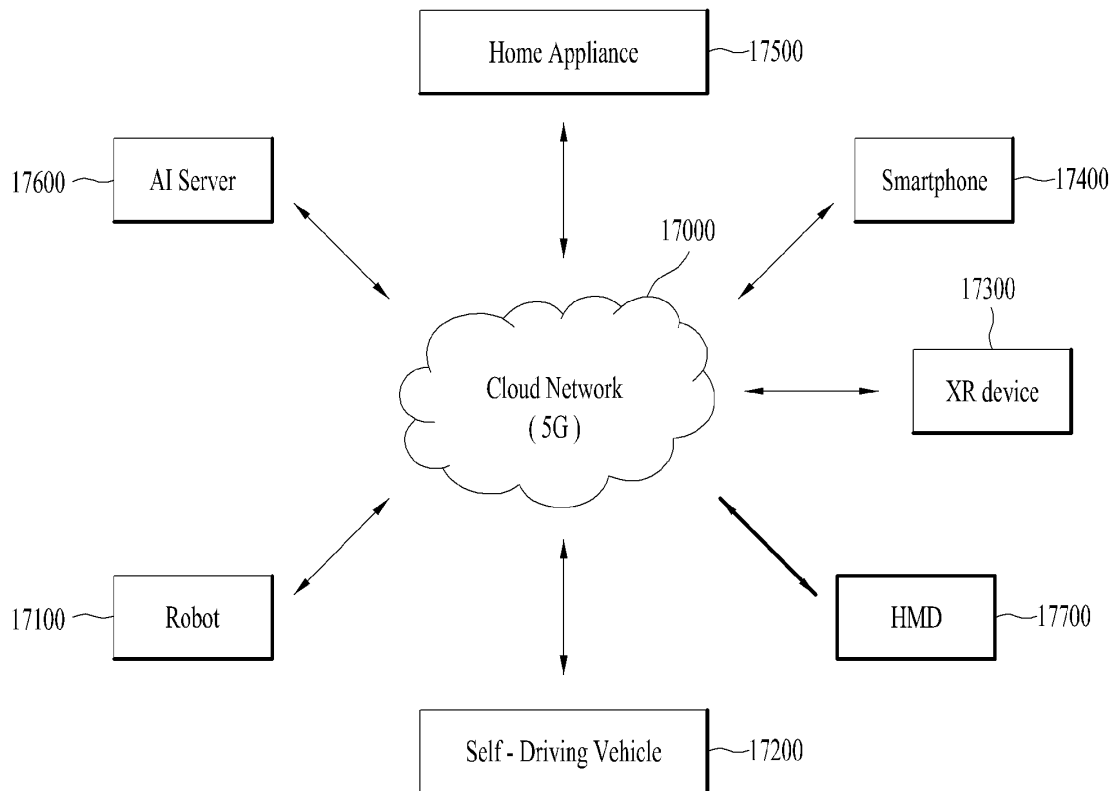
[도 12]



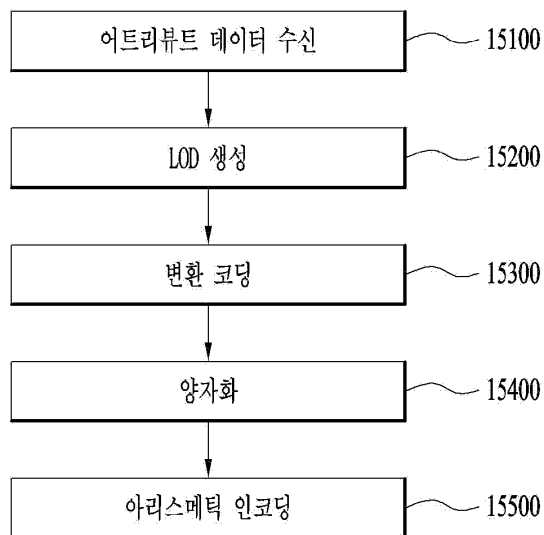
[도 13]



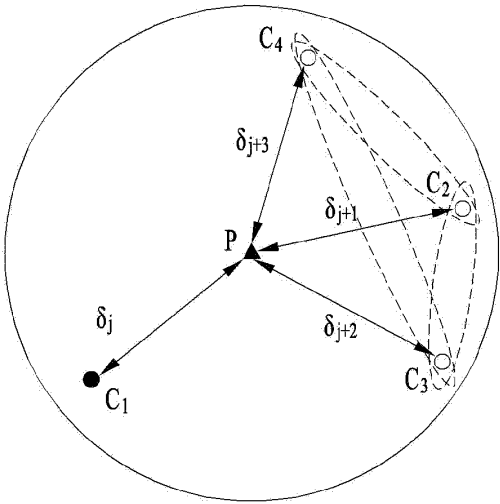
[도 14]



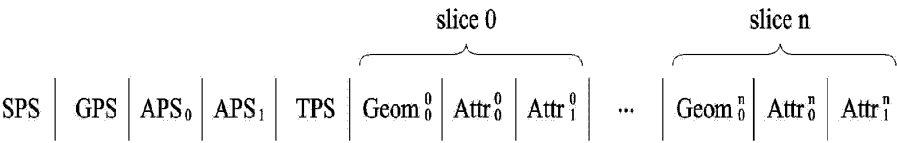
[도 15]



[도16]



[도17]



[도 18]

```

seq_parameter_set() {
    profile_compatibility_flags
    level_idc
    sps_bounding_box_present_flag
    if( sps_bounding_box_present_flag ) {
        sps_bounding_box_offset_x
        sps_bounding_box_offset_y
        sps_bounding_box_offset_z
        sps_bounding_box_scale_factor
        sps_bounding_box_size_width
        sps_bounding_box_size_height
        sps_bounding_box_size_depth
    }
    sps_source_scale_factor
    sps_seq_parameter_set_id
    sps_num_attribute_sets
    for( i=0; i<sps_num_attribute_sets; i++ ) {
        attribute_dimension[i]
        attribute_instance_id[i]
        attribute_bitdepth[i]
        attribute_cicp_colour_primaries[i]
        attribute_cicp_transfer_characteristics[i]
        attribute_cicp_matrix_coeffs[i]
        attribute_cicp_video_full_range_flag[i]
        attribute_correlated_weight_flag
        attribute_correlated_weight_method
        known_attribute_label_flag[i]
        if( known_attribute_label_flag[i] )
            known_attribute_label[i]
        else
            attribute_label_four_bytes[i]
    }
    sps_extension_present_flag
    if( sps_extension_present_flag )
        while( more_data_in_byte_stream( ) )
            sps_extension_data_flag
    byte_alignment( )
}

```

[19]

attribute_parameter_set() {
aps_attr_parameter_set_id
aps_seq_parameter_set_id
attr_coding_type
aps_attr_initial_qp
aps_attr_chroma_qp_offset
aps_slice_qp_delta_present_flag
aps_correlated_weight_flag
aps_correlated_weight_method
isLifting = (attr_coding_type == 0 attr_coding_type == 2) ? 1 : 0
if (isLifting) {
lifting_num_pred_nearest_neighbours
lifting_max_num_direct_predictors
lifting_search_range
lifting_lod_regular_sampling_enabled_flag
lifting_num_detail_levels_minus1
for(idx = 0; idx <= num_detail_levels_minus1; idx ++) {
if (lifting_lod_decimation_enabled_flag)
lifting_sampling_period[idx]
else
lifting_sampling_distance_squared[idx]
}
}
if(attr_coding_type == 0)
lifting_adaptive_prediction_threshold
lifting_intra_lod_prediction_num_layers
}
aps_extension_present_flag
if(aps_extension_present_flag)
while(more_data_in_byte_stream())
aps_extension_data_flag
byte_alignment()
}

[도20]

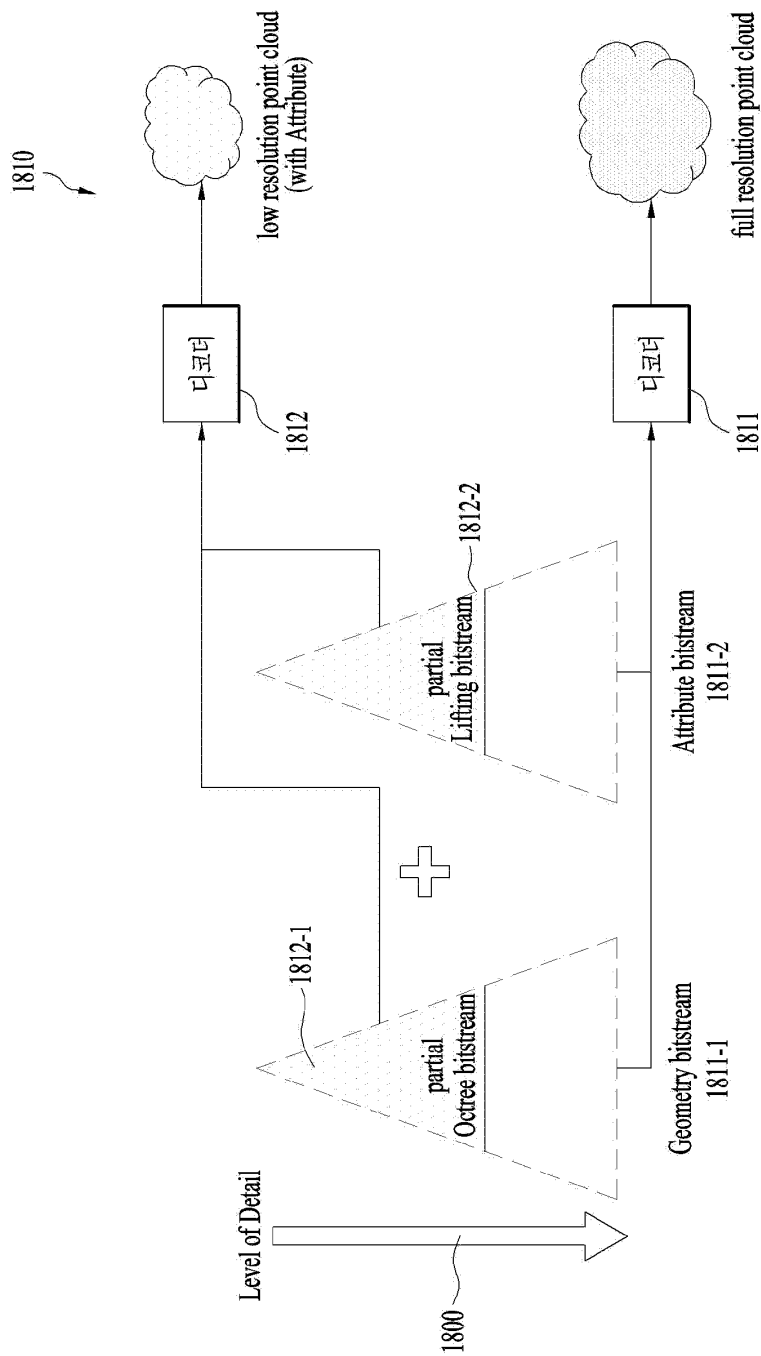
```
//pd: point distance  
weighted_sum = alpha*pd0 + beta*pd1 + gamma*pd2;  
w2 = alpha*pd0 / weighted_sum;  
w1 = beta*pd1 / weighted_sum;  
w0 = reference_power - (w1 + w2);
```

```
//pd: point distance  
weighted_sum = pd0 + pd1 + pd2;  
w2 = reference_power*(pd0 / weighted_sum);  
w1 = reference_power*(pd1 / weighted_sum);  
w0 = reference_power - (w1 + w2);
```

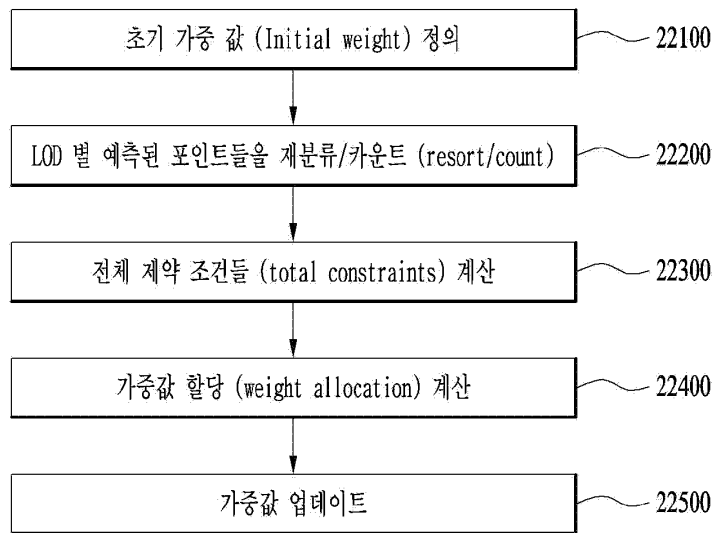
```
//pd: point distance  
weighted_sum = pd0*pd0 + pd1*pd1 + pd2*pd2;  
w2 = pd0*pd0 / weighted_sum;  
w1 = pd1*pd1 / weighted_sum;  
w0 = reference_power - (w1 + w2);
```

```
//pd: point distance  
weighted_sum = pd0+pd0 + pd1+pd1 + pd2+pd2;  
w2 = pd0+pd0 / weighted_sum;  
w1 = pd1+pd1 / weighted_sum;  
w0 = reference_power - (w1 + w2);
```

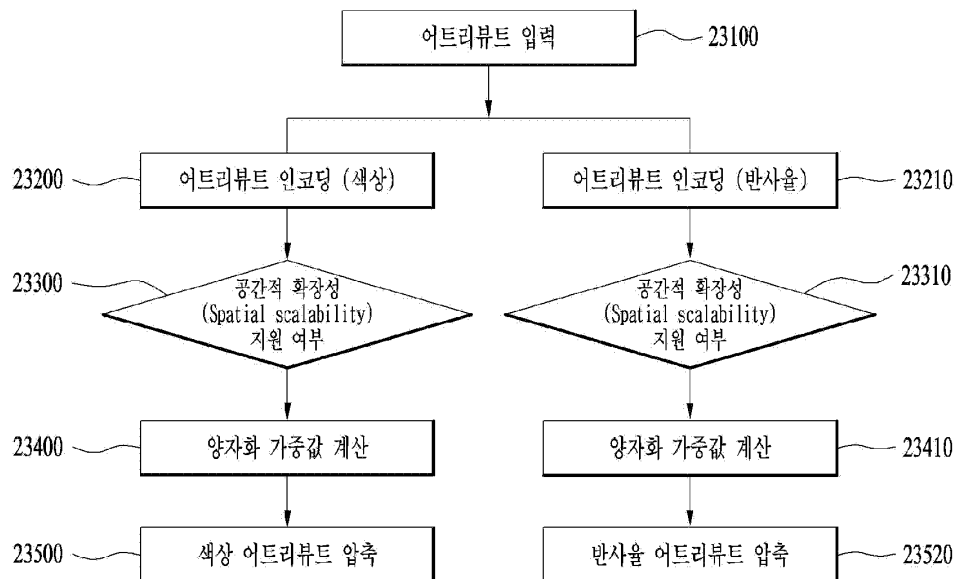
[도21]



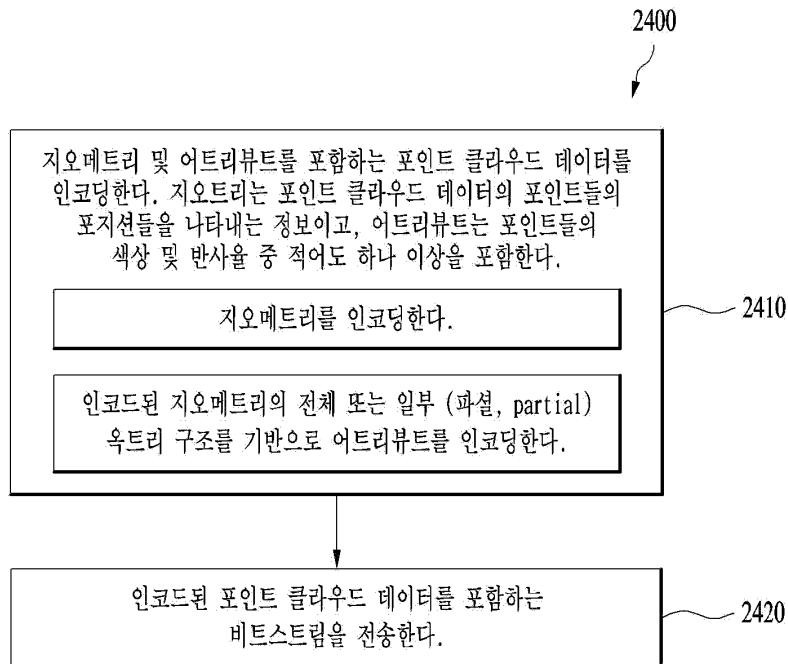
[도22]



[도23]



[도24]



[도25]

