*ALE

The Arbitrary-Lagrangian-Eulerian, or ALE, method is an explicit Finite Element hydrocode. It handles multiple materials (multi-materials) in a single ALE mesh. ALE meshes are neither fixed in space nor attached to a material. The ALE mesh(es) may transform (translate, rotate, and expand) as needed. These transformations can reduce the number of elements required, potentially improving computational efficiency.

A model can contain more than one ALE mesh. Within each ALE mesh multiple ALE materials may move around, deform, and impact each other. If some Lagrangian structures (meshes) overlap with the ALE domain, the ALE materials can interact with these structures. These structures may be constructed as any combination of Lagrangian, SPH, SPG and DEM parts. This fluid-structure interaction (FSI) capability enables analyzing very complex FSI problems. The ALE solver does not have turbulence or boundary layer features, so it is not intended for modeling the traditional CFD problems which require those features.

Generally, the ALE mesh is unstructured to accommodate complex geometries. For simple rectilinear geometries, a structured, logically regular, mesh can be used with the Structure ALE (S-ALE) solver. This solver has algorithmic simplifications, memory reductions, and performance enhancements due to the type of mesh.

ALE *does not* support implicit time integration, nor does it support dynamic relaxation. Furthermore, except for ALE formulation 5, which *does* support contact, ALE *does not*, in general, support contact.

In three dimensions, ALE supports *only* one-point solid elements. These solid elements can either be hexahedral, pentahedral, or tetrahedral. Pentahedrons and tetrahedrons are treated as degenerate hexahedron elements. For each ALE multi-material, strain and stress is evaluated in each solid element at a single integration point. In this sense, the ALE element formulation is equivalent to ELEFORM 1 solid formulation.

Input required for LS-DYNA's ALE capability is specified using *ALE cards. The keyword cards in this section, excluding those specifically for the S-ALE solver, are listed in alphabetical order:

*ALE_AMBIENT_HYDROSTATIC

*ALE_BURN_SWITCH_MMG

*ALE_COUPLING_NODAL_CONSTRAINT

*ALE_COUPLING_NODAL_DRAG

*ALE_COUPLING_NODAL_PENALTY

*ALE_COUPLING_RIGID_BODY

*ALE_ESSENTIAL_BOUNDARY

*ALE_FAIL_SWITCH_MMG

*ALE_FRAGMENTATION

*ALE_FSI_PROJECTION

ALE_FSI_SWITCH_MMG_*{OPTION}

*ALE_FSI_TO_LOAD_NODE

*ALE_INJECTION

*ALE_MAPPING

*ALE_MAPPING_FROM_LAGRANGIAN

*ALE_MESH_INTERFACE

*ALE_MULTI-MATERIAL_GROUP

*ALE_PRESCRIBED_MOTION

*ALE_REFERENCE_SYSTEM_CURVE

*ALE_REFERENCE_SYSTEM_GROUP

*ALE_REFERENCE_SYSTEM_NODE

*ALE_REFERENCE_SYSTEM_SWITCH

*ALE_REFINE

*ALE_SMOOTHING

*ALE_SWITCH_MMG

*ALE_TANK_TEST

*ALE_UP_SWITCH

Keywords specifically for the S-ALE solver are:

*ALE_STRUCTURED_FSI

*ALE_STRUCTURED_MESH

*ALE_STRUCTURED_MESH_CONTROL_POINTS

*ALE_STRUCTURED_MESH_JOIN

*ALE_STRUCTURED_MESH_MOTION

*ALE_STRUCTURED_MESH_REFINE

*ALE_STRUCTURED_MESH_TRIM

*ALE_STRUCTURED_MESH_VOLUME_FILLING

*ALE_STRUCTURED_MULTI-MATERIAL_GROUP

*ALE_STRUCTURED_POINT_SOURCE

For simplifying applying boundary conditions to S-ALE meshes, see:

*BOUNDARY_SALE_MESH_FACE

For other input information related to the ALE capability, see keywords:

*BOUNDARY_AMBIENT_EOS

*CONSTRAINED_EULER_IN_EULER

*CONSTRAINED_LAGRANGE_IN_SOLID

*CONTROL_ALE

*DATABASE_FSI

*INITIAL_VOID

*INITIAL_VOLUME_FRACTION

*INITIAL_VOLUME_FRACTION_GEOMETRY

*SECTION_SOLID

*SECTION_POINT_SOURCE_FOR_GAS_ONLY

*SECTION_POINT_SOURCE_MIXTURE

*SET_MULTIMATERIAL_GROUP_LIST

For a single gaseous material:

*EOS_LINEAR_POLYNOMIAL

*EOS_IDEAL_GAS

*MAT_NULL

For multiple gaseous materials:

*MAT_GAS_MIXTURE

*INTIAL_GAS_MIXTURE

*ALE_AMBIENT_HYDROSTATIC

Purpose: When an ALE model contains one or more ambient (or reservoir-type) ALE parts (ELFORM = 11 and AET = 4), this command may be used to initialize the hydrostatic pressure field in the ambient ALE domain due to gravity. The *LOAD_BODY_*OPTION* keyword must be defined. The associated *INITIAL_HYDROSTATIC_ALE keyword may be used to define a similar initial hydrostatic pressure field for the regular ALE domain (not reservoir-type region).

Card 1	1	2	3	4	5	6	7	8
Variable	ALESID	STYPE	VECID	GRAV	PBASE	RAMPTLC		
Type	I	I	I	F	F	I		
Default	none	0	none	0.0	0.0	0		

Card 2	1	2	3	4	5	6	7	8
Variable	NID	MMGBL						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
ALESID	ALESID defines the reservoir-type. ALE domain/mesh whose hydrostatic pressure field due to gravity is being initialized by this keyword. See Remark 4.
STYPE	ALESID set type. See Remark 4. EQ.0: Part set ID (PSID), EQ.1: Part ID (PID), EQ.2: Solid set ID (SSID).
VECID	Vector ID of a vector defining the direction of gravity.

VARIABLE	DESCRIPTION
GRAV	Magnitude of the Gravitational acceleration. For example, in metric units the value is usually set to 9.80665 m/s^2.
PBASE	Nominal or reference pressure at the top surface of all fluid layers. By convention, the gravity direction points from the top layer to the bottom layer. Each fluid layer must be represented by an ALE multi-material group ID (AMMGID or MMG). See Remark 1.
RAMPTLC	A ramping time function load curve ID. This curve (via *DEFINE_CURVE) defines how gravity is ramped up as a function of time. Given GRAV value above, the curve's ordinate varies from 0.0 to 1.0, and its abscissa is the (ramping) time. See Remark 2.
NID	Node ID defining the top of an ALE fluid (AMMG) layer.
MMGBL	AMMG ID of the fluid layer immediately below this NID. Each node is defined in association with one AMMG layer below it. See Remark 3. In case of S-ALE, AMMG name (AMMGNM) could be also used in place of AMMGID. See Remark 5.

Remarks:

1. **Pressure in Multi-Layer Fluids.** For models using multi-layer ALE Fluids the pressure at the top surface of the top fluid layer is set to PBASE and the hydrostatic pressure is computed as following

$$P = P_{\text{base}} + \sum_{i=1}^{N_{\text{layers}}} \rho_i g h_i \, .$$

2. **Hydrostatic Pressure Ramp Up.** If RAMPTLC is activated (i.e. not equal to "0"), then the hydrostatic pressure is effectively ramped up over a user-defined duration and kept steady. When this load curve is defined, do not define the associated *INITIAL_HYDROSTATIC_ALE card to initialize the hydrostatic pressure for the non-reservoir ALE domain. The hydrostatic pressure in the regular ALE region will be initialized indirectly because of the hydrostatic pressure generated in the reservoir-type ALE domain. The same load curve should be used to ramp up gravity in a corresponding *LOAD_BODY card. With this approach, any submerged Lagrangian structure coupled to the ALE fluids will have time to equilibrate to the proper hydrostatic condition.

3. **Limitation on EOS Model.** This keyword only supports *EOS_GRUNEISEN and *EOS_LINEAR_POLYNOMIAL, but only in the following two cases:

$$c_3 = c_4 = c_5 = c_6 = 0, \qquad E_0 = 0$$
$$c_4 = c_5 > 0, \qquad c_1 = c_2 = c_3 = c_6 = 0, \qquad V_0 = 0.$$

4. **Structured ALE usage.** When used with structured ALE, the PART and PART set options might not make too much sense. This is because all elements inside a structured ALE mesh are assigned to one single PART ID. In the Structured ALE case, we should generate a solid set which contains those ALE boundary elements we want to prescribe hydrostatic pressures on. To do this, define the solid set using *SET_SOLID_GENERAL keyword with the SALECPT option and set STYPE = 2 (Solid element set ID) for this keyword.

5. **AMMG NAME for S-ALE.** For the general ALE solver, you define each AMMG with *ALE_MULTI-MATERIAL_GROUP. In this case, each AMMG can only be referred to by their AMMGID. The AMMGID for each AMMG is based on the order of appearance of the AMMG in the input deck. For the S-ALE solver, you can define the AMMG using *ALE_STRUCTURED_MULTI-MATERIAL_-GROUP instead of *ALE_MULTI-MATERIAL_GROUP. With *ALE_STRUC-TURED_MULTI-MATERIAL_GROUP, you give each AMMG a name with the field AMMGNM. Each AMMG defined with that keyword can then be referred with either its name or its AMMGID (which is again based on order of appear-ance). We recommend using the name as it leads to fewer errors. For instance, if you add or delete AMMGs, then the AMMGIDs may change. Then, you must find all those references and change them accordingly. With the name, you do not need to modify the input deck for unchanged AMMGs.

Example:

Model Summary: Consider a model consisting of 2 ALE parts, air on top of water.

H3 = AMMG1 = Air part above

H4 = AMMG2 = Water part below

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
$ ALE materials (fluids) listed from top to bottom:
$
$ NID AT TOP OF A LAYER SURFACE         ALE MATERIAL LAYER BELOW THIS NODE
$ TOP OF 1st LAYER -------> 1681        ----------------------------------------
$                                       Air above   = PID 3 = H3 = AMMG1 (AET=4)
$ TOP OF 2nd LAYER -------> 1671        ----------------------------------------
$                                       Water below = PID 4 = H4 = AMMG2 (AET=4)
$ BOTTOM ---------------------->        ----------------------------------------
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_AMBIENT_HYDROSTATIC
$   ALESID     STYPE     VECID      GRAV     PBASE   RAMPTLC
        34         0        11   9.80665  101325.0         9
$      NID    MMGBL
      1681         1
      1671         2
*SET_PART_LIST
```

```
        34
         3         4
*ALE_MULTI-MATERIAL_GROUP
         3         1
         4         1
*DEFINE_VECTOR
$      VID        XT        YT        ZT        XH        YH        ZH       CID
        11       0.0       1.0       0.0       0.0       0.0       0.0
*DEFINE_CURVE
         9
               0.000             0.000
               0.001             1.000
              10.000             1.000
*LOAD_BODY_Y
$     LCID        SF    LCIDDR        XC        YC        ZC
         9   9.80665         0       0.0       0.0       0.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_BURN_SWITCH_MMG

Purpose: Change a volume fraction of an explosive modeled by an ALE multi-material group (AMMGID) into the explosion product (another ALE group). The reaction starts if some ignition conditions are met. These conditions and the reaction rate computing the explosive fraction to be switched are specified by *DEFINE_FUNCTION functions. The functions take as many arguments as there are fields specified on Card 4.

Card Summary:

Card 1. This card is required.

MMGFR	MMGTO	NVARLINE					

Card 2. This card is required.

REACT							

Card 3. This card is required.

IGNI	IGNIV	IGNIVF					

Card 4. Include NVARLINES of this card.

VAR	VAR	VAR	VAR	VAR	VAR	VAR	VAR

Card 5. Include as many of this card as desired. The next keyword ("*") card terminates this input.

PAR	PAR	PAR	PAR	PAR	PAR	PAR	PAR

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	MMGFR	MMGTO	NVARLINE					
Type	I	I	I					
Default	none	none	0					

VARIABLE	DESCRIPTION
MMGFR	ALE multi-material-group (explosive) before the switch
MMGTO	ALE multi-material-group (explosion product) after the switch
NVARLINE	Number of lines with arguments in the functions REACT, IGNI and IGNIV

Card 2	1	2	3	4	5	6	7	8
Variable	REACT							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
REACT	ID of the *DEFINE_FUNCTION function controlling the reaction rate. This function determines the explosive volume fraction to be switched.

Card 3	1	2	3	4	5	6	7	8
Variable	IGNI	IGNIV	IGNIVF					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
IGNI	ID of the *DEFINE_FUNCTION function controlling the conditions of ignition
IGNIV	ID of the *DEFINE_FUNCTION function computing the ignition front speed. See Remark 1.
IGNIVF	Flag that activates computing the ignition front as a material interface between MMGFR and MMGTO. This flag will automatically

VARIABLE	DESCRIPTION
	turn on if *both* IGNI and IGNIVF are undefined (see Remark 2). EQ.0: Not activated EQ.1: Activated

Variable Cards. Cards defining the function arguments. Include NVARLINES of this card.

Card 4	1	2	3	4	5	6	7	8
Variable	VAR1	VAR2	VAR3	VAR4	VAR5	VAR6	VAR7	VAR8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
VAR*i*	Variables that provide the arguments for REACT, IGNI, and IGNIV (see Remark 3): EQ.1: *xx*-stress for MMGFR EQ.2: *yy*-stress for MMGFR EQ.3: *zz*-stress for MMGFR EQ.4: *xy*-stress for MMGFR EQ.5: *yz*-stress for MMGFR EQ.6: *zx*-stress for MMGFR EQ.7: Plastic strain for MMGFR EQ.8: Internal energy for MMGFR EQ.9: Bulk viscosity for MMGFR EQ.10: Volume from the previous cycle for MMGFR GE.11 and LE.20: Other auxiliary variables for MMGFR GE.21 and LE.40: Auxiliary variables for MMGTO (*xx*-stress, …) EQ.41: Mass for MMGFR

VARIABLE	DESCRIPTION
EQ.42:	Mass for MMGTO
EQ.43:	Volume fraction for MMGFR
EQ.44:	Volume fraction for MMGTO
EQ.45:	Material volume for MMGFR
EQ.46:	Material volume for MMGTO
EQ.47:	Time step
EQ.48:	Time
EQ.49:	Cycle
GE.50 and LE.57:	x-positions of the ALE nodes
GE.58 and LE.65:	y-positions of the ALE nodes
GE.66 and LE.73:	z-positions of the ALE nodes
GE.74 and LE.81:	x-velocities of the ALE nodes
GE.82 and LE.89:	y-velocities of the ALE nodes
GE.90 and LE.97:	z-velocities of the ALE nodes
GE.98 and LE.105:	x-accelerations of the ALE nodes
GE.106 and LE.113:	y-accelerations of the ALE nodes
GE.114 and LE.121:	z-accelerations of the ALE nodes
GE.122 and LE.129:	Masses of the ALE nodes

Parameter Cards. Cards defining parameters to control user-defined routines in the user-subroutine Fortran files:

- `alebrnswmg_ignitioncondition`
- `alebrnswmg_ignitionspeed`
- `alebrnswmg_reactionrate`

They can be used along with or instead of the functions IGNI, IGNIV, and REACT. The usermat version of LS-DYNA is required to access these routines. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 5	1	2	3	4	5	6	7	8
Variable	PAR1	PAR2	PAR3	PAR4	PAR5	PAR6	PAR7	PAR8
Type	F	F	F	F	F	F	F	F
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
PAR*i*	User-defined routines parameters

Remarks:

1. **Nodal lighting times.** Each ALE node has a lighting time. When reached for a given node *i*, the reaction in the elements connected to this node can begin. The lighting time of another node (*j*) of an element connected to node *i* is computed by dividing the distance from *j* to *i* by the ignition speed provided by IGNIV. The first nodes to be ignited can be selected with *INITIAL_DETONATION. If the detonation point (located by (X, Y, Z) in *INITIAL_DETONATION) is close to a node, this node becomes the first node to be ignited. If the detonation point is close to an element center, the nodes of this element are first ignited.

2. **Ignition front.** If IGNIVF = 1, the ignition front is a material interface between MMGFR and MMGTO. Every node behind the front is ignited; if a node in an element with MMGFR and MMGTO is on the MMGTO side, its lighting time is set to the current time to activate the reaction in the elements connected to this node.

3. **Variable specification.** The variables are presented to the function IDFUNC as floating point data. The order of the arguments appearing in the *DEFINE_-FUNCTION should match the order of VAR*i* specified on Card 4 (for this

keyword). For example, when there is one card in format 4 containing "48, 49, 41, 42", then the time (48), the cycle (49), and the masses (41 & 42) should be the first, second, third, and fourth arguments to the function defined on the *DEFINE_FUNCTION keyword.

If there is a blank column between 2 VAR*i* fields, the list between the values in these two fields is specified. For example, if the card contains "1, ,6", then the six stresses (1 through 6) are selected as arguments.

*ALE_COUPLING_NODAL_CONSTRAINT_{OPTION}

Available options include:

<BLANK>

ID

TITLE

Purpose: This keyword activates constraint coupling between ALE materials and non-ALE (structure) nodes. The structure nodes may belong to Lagrangian solid, shell, beam, thick shell, or discrete sphere (see *ELEMENT_DISCRETE_SPHERE) elements. In contrast to *ALE_COUPLING_NODAL_PENALTY, caution should be exercised when using EFG, SPG or SPH nodes, as meshless methods generally do not satisfy essential boundary conditions, leading to energy dissipation.

> This keyword requires a 3D ALE formulation. It is, therefore, incompatible with parts defined using *SECTION_ALE2D or *SECTION_ALE1D.

If a title is not defined, LS-DYNA will automatically create an internal title for this coupling definition.

Title Card. Additional card for TITLE and ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	STRSID	ALESID	STRSTYP	ALESTYP	CTYPE	MCOUP		
Type	I	I	I	I	I	I		
Default	none	none	0	0	1	0		

Card 2	1	2	3	4	5	6	7	8
Variable	START	END				FRCMIN		
Type	F	F				F		
Default	0	10^{10}				0.5		

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number (I10). If not defined, LSDYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	A description of this coupling definition (A70)
STRSID	ID for a part, part set, segment set, or node set giving the structure. The structure may include Lagrangian solid, shell, beam, thick shell, or discrete sphere elements. EFG, SPH, or EFG nodes may be used, but the boundary conditions may not be satisfied.
ALESID	ID for a part or part set specifying the ALE solid elements
STRSTYP	Set type for STRSID: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART) EQ.2: Segment set ID (see *SET_SEGMENT) EQ.3: Node set ID (see *SET_NODE)
ALESTYP	Set type for ALESID: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART)
CTYPE	Coupling type: EQ.1: Constrained velocity only EQ.2: Constrained acceleration and velocity
MCOUP	Multi-material option (see Remarks). EQ.0: Couple with all multi-material groups

VARIABLE	DESCRIPTION
	LT.0: MCOUP must be an integer. –MCOUP refers to a set ID of an ALE multi-material groups defined in *SET_MULTI-MATERIAL_GROUP.
START	Start time for coupling
END	End time for coupling
FRCMIN	Only to be used with nonzero MCOUP. Minimum volume fraction of the fluid materials included in the list of AMMGs to activate coupling. Default value is 0.5.

Remarks:

When MCOUP is a negative integer, say for example MCOUP = -123, then an ALE multi-material set-ID (AMMSID) of 123 must exist. This is an ID defined by a *SET_MULTI-MATERIAL_GROUP_LIST card.

*ALE_COUPLING_NODAL_DRAG

Available options include:

<BLANK>

ID

TITLE

Purpose: This command provides a coupling mechanism to model the drag interaction between ALE fluids and discrete element forms (particles). The particles are assumed to be of spherical shape being either SPH elements or discrete elements (*ELEMENT_DISCRETE_SPHERE). The coupling forces are proportional to the relative speed between the fluid and particles plus the buoyancy force due to gravitational loading.

> This keyword requires a 3D ALE formulation. It is, therefore, incompatible with parts defined using *SECTION_ALE2D or *SECTION_ALE1D.

If the title is not defined, LS-DYNA will automatically generate an internal title for this coupling definition.

Title Card. Additional card for TITLE and ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	STRSID	ALESID	STRSTYP	ALESTYP				
Type	I	I	I	I				
Default	none	none	0	0				

Card 2	1	2	3	4	5	6	7	8
Variable	START	END		FCOEF			DIRECG	GRAV
Type	F	F		F			I	F
Default	0	10^{10}		1.0			none	0.0

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number (I10). If not defined, LS-DYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	A description of this coupling definition (A70).
STRSID	ID for a part, part set, segment set, or node set specifying the particles. The particles can be SPH or discrete elements.
ALESID	ID for a part or part set ID giving the ALE solid elements. See Remark 1.
STRSTYP	Particle set type: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART) EQ.2: Segment set ID (see *SET_SEGMENT) EQ.3: Node set ID (see *SET_NODE)
ALESTYP	ALE set type: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART)
START	Start time for coupling
END	End time for coupling
FCOEF	Drag coefficient scale factor or function ID to calculate drag coefficient GT.0: Drag coefficient scale factor.

VARIABLE	DESCRIPTION
	LT.0: The absolute value of FCOEF is the Function ID of the user provided function to calculate drag coefficient; See Remark 1.
DIRECG	Gravity force direction. EQ.1: Global x direction EQ.2: Global y direction EQ.3: Global z direction
GRAV	Gravity value. This value is used to calculate buoyance force.

Remarks:

1. **Drag Coupling Force.** The drag coupling force between the particles and ALE fluids takes the following form:

$$F_{\text{drag}} = c_{\text{drag}} \times \frac{1}{2}\rho v^2 \times \frac{1}{4}\pi d^2 \ ,$$

where c_{drag} is the drag coefficient, ρ is the density of the fluid in which the particle is submerged, v is the relative velocity between the particle and the fluid, and d is the diameter of the particle.

The default drag coefficient is a function of Reynolds's number and calculated by using the following formula:

$$c_{\text{drag}} = \left(0.63 + \frac{4.8}{\sqrt{\text{Re}}}\right)^2.$$

You can define your own function of drag coefficient. To do that, you need to define a function using *DEFINE_FUNCTION and assign the negative function ID to FCOEF flag.

An example is provided below to illustrate the setup. It is equivalent to the default drag coefficient calculation.

```
*ALE_COUPLING_NODAL_DRAG
     10001         1         3         1
                                    -10                                 3      9.81
*DEFINE_FUNCTION
        10
float cd(float re)
{
  float cd;
  cd=(0.63+4.8/sqrt(re))*(0.63+4.8/sqrt(re));
  if (cd > 2.0) cd = 2.0;
  return cd;
}
```

*ALE_COUPLING_NODAL_PENALTY

Available options include:

<BLANK>

ID

TITLE

Purpose: This command provides a penalty coupling mechanism between ALE materials and non-ALE (structure) nodes. The structure nodes may belong to Lagrangian solid, shell, beam, thick shell, or discrete (*ELEMENT_DISCRETE_SPHERE) elements. In contrast to *ALE_COUPLING_NODAL_CONSTRAINT, EFG, SPG, and SPH nodes *are* supported.

> This keyword is incompatible with parts that use *SECTION_ALE2D or *SECTION_ALE1D, meaning it requires a 3D ALE formulation.

If a title is not defined, LS-DYNA will automatically create an internal title for this coupling definition.

Title Card. Additional card for TITLE and ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	STRSID	ALESID	STRSTYP	ALESTYP		MCOUP		
Type	I	I	I	I		I		
Default	none	none	0	0		0		

Card 2	1	2	3	4	5	6	7	8
Variable	START	END	PFORM	PFAC		FRCMIN		
Type	F	F	I	F		F		
Default	0	10^{10}	0	0.1		0.5		

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID number (I10). If not defined, LSDYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	A description of this coupling definition (A70)
STRSID	ID of a part, part set, segment set, or node set giving the structure. The structure may include Lagrangian elements, EFG, SPG, or SPH.
ALESID	ID of a part or part set specifying the ALE solid elements
STRSTYP	Set type for STRSID: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART) EQ.2: Segment set ID (see *SET_SEGMENT) EQ.3: Node set ID (see *SET_NODE)
ALESTYP	Set type of ALESID: EQ.0: Part set ID (see *SET_PART) EQ.1: Part ID (see *PART)
MCOUP	Multi-material option (see Remarks): EQ.0: Couple with all multi-material groups LT.0: MCOUP must be an integer. –MCOUP refers to a set ID of an ALE multi-material groups defined in *SET_MULTI-MATERIAL_GROUP.
START	Start time for coupling

VARIABLE	DESCRIPTION
END	End time for coupling
PFORM	Penalty stiffness formulations: EQ.0: Mass based penalty stiffness EQ.1: Bulk modulus based penalty stiffness EQ.2: Penalty stiffness is determined by the user-provided load curve between penetration and penalty pressure.
PFAC	Penalty stiffness factor (PFORM = 0 or 1) for scaling the estimated stiffness of the interacting (coupling) system or load curve ID (PFORM = 2).
FRCMIN	Only to be used with nonzero MCOUP. Minimum volume fraction of the fluid materials included in the list of AMMGs to activate coupling. Default value is 0.5.

Remarks:

When MCOUP is a negative integer, say for example MCOUP = -123, then an ALE multi-material set-ID (AMMSID) of 123 must exist. This is an ID defined by a *SET_MULTI-MATERIAL_GROUP_LIST card.

*ALE_COUPLING_RIGID_BODY

Purpose: This command serves as a simplified constraint type coupling method between ALE fluids and a Lagrange rigid body.

In certain FSI simulations structure deformation is either small or not of the interest. Often these structures are modeled as rigid bodies to shorten the simulation time and reduce the complexity. For such kind of problems, a full scale ALE/FSI simulation is costly in both simulation time and memory. This keyword provides a light-weight alternative FSI method for systems with minimal structural response.

It has a similar input format to *ALE_ESSENTIAL_BOUNDARY and maybe regarded as being an extension of the essential boundary feature. The documentation for *ALE_ESSENTIAL_BOUNDARY_BODY applies, in large part, to this card also.

Card 1	1	2	3	4	5	6	7	8
Variable	PID	NSID						
Type	I	I						
Default	none	none						

ALE Coupling Interfaces Cards. Include one card for each part, part set or segment to define ALE coupling interface. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	ID	IDTYPE	ICTYPE	IEXCL				
Type	I	I	I	I				
Default	none	none	1	none				

VARIABLE	DESCRIPTION
PID	Rigid body part ID
NSID	Node set ID defining ALE boundary nodes to follow rigid body motion

VARIABLE	DESCRIPTION
ID	Set ID defining a part, part set or segment set ID of the ALE coupling interface
IDTYPE	Type of set ID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID) EQ.2: Segment set ID (SGSID)
ICTYPE	Constraint type: EQ.1: No flow through all directions EQ.2: No flow through normal direction (slip condition)
IEXCL	Segment Set ID to be excluded from applying ALE essential boundary condition. For example, inlet/outlet segments.

Remarks:

For ICTYPE = 2, the constrained direction(s) at each surface node comes in part from knowing whether the node is a surface node, an edge node, or a corner node. If the ALE mesh boundary is identified by part(s) (IDTYPE = 0/1), edge and corner nodes are automatically detected during the segment generation process. However, this automatic detection is not foolproof for complicated geometries. Identifying the ALE mesh boundary using segment sets (IDTYPE = 2) is generally preferred for complicated geometries in order to avoid misidentification of edge and corner nodes. When segment sets are used, the edge and corner nodes are identified by their presence in multiple segment sets where each segment set describes a more or less smooth, continuous surface. The intersections of these surfaces are used to identify edge/corner nodes.

*ALE_ESSENTIAL_BOUNDARY

Purpose: This command applies and updates essential boundary conditions on ALE or S-ALE boundary surface nodes. Updating the boundary conditions is important if the ALE mesh moves according to *ALE_REFERENCE_SYSTEM_GROUP. If the mesh does not move, it's more correct to call it an Eulerian mesh rather than an ALE mesh, but *ALE_ESSENTIAL_BOUNDARY can be applied nonetheless.

Certain engineering problems need to constrain the flow along the ALE mesh boundary. A simple example would be water flowing in a curved tube. Using the *ALE_ESSENTIAL_BOUNDARY approach, the tube material is not modeled and there is no force coupling between the fluid and the tube, rather the interior volume of the tube is represented by the location of the ALE mesh. Defining SPC boundary conditions with a local coordinate system at each ALE boundary node would be extremely inconvenient in such a situation. The *ALE_ESSENTIAL_BOUNDARY command applies the desired constraints along the ALE surface mesh automatically. You only need to specify the part(s) or segment set(s) corresponding to the ALE boundary surfaces and the type of constraint desired.

Boundary Condition Cards. Include one card for each part, part set or segment on which essential boundary conditions are applied. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID	IDTYPE	ICTYPE	IEXCL				
Type	I	I	I	I				
Default	none	none	1	none				

VARIABLE	DESCRIPTION
ID	Set ID defining a part, part set or segment set ID of the ALE or S-ALE mesh boundary.
IDTYPE	Type of set ID: EQ.0: Part set ID (PSID). EQ.1: Part ID (PID). EQ.2: Segment set ID (SGSID).

VARIABLE	DESCRIPTION
ICTYPE	Constraint type: EQ.1: No flow through all directions. EQ.2: No flow through normal direction. (slip condition)
IEXCL	Segment set ID to be excluded from applying ALE or S-ALE essential boundary condition. For example, inlet/outlet segments.

Remarks:

For ICTYPE = 2, the constrained direction(s) at each surface node comes in part from knowing whether the node is a surface node, an edge node, or a corner node. If the ALE mesh boundary is identified by part(s) (IDTYPE = 0 or 1), edge/corner nodes are automatically detected during the segment generation process. However, this automatic detection is not foolproof for complicated geometries. Identifying the ALE mesh boundary using segment sets (IDTYPE = 2) is generally preferred for complicated geometries in order to avoid misidentification of edge/corner nodes. When segment sets are used, the edge/corner nodes are identified by their presence in multiple segment sets where each segment set describes a more or less smooth, continuous surface. In short, the junctures or intersections of these surfaces identify edge/corner nodes.

*ALE_FAIL_SWITCH_MMG_{OPTION}

Purpose: This card is used to allow the switching of an ALE multi-material-group ID (AMMGID) if a failure criteria is reached. If this card is not used and *MAT_VACUUM has a multi-material group in the input deck, failed ALE groups are replaced by the group for *MAT_VACUUM.

Available options include:

<BLANK>

ID

TITLE

A title for the card may be input between the 11th and 80th character on the title-ID line. The optional title line precedes all other cards for this command.

The user can explicitly define a title for this coupling.

Title Card. Additional card for the ID or TITLE options to keyword.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	FR_MMG	TO_MMG						
Type	I	I						
Default	none	none						

VARIABLE	DESCRIPTION
FR_MMG	This is the AMMG-SID before the switch. The AMMG-SID corresponds to the SID defined under the *SET_MULTI-MATERIAL_GROUP_LIST (SMMGL) card. This SID points to one or more AMMGs. See Remark 1.

VARIABLE	DESCRIPTION
TO_MMG	This is the AMMG-SID after the switch. The AMMG-SID corresponds to the SID defined under the *SET_MULTI-MATERIAL_GROUP_LIST card. This SID points to one or more AMMGs. See Remark 1.

Remarks:

1. **AMMG Switch.** There is a correspondence between the FR_MMG and TO_MMG. Consider an example where:

   a) The FR_MMG SID points to a SID = 12 (the SID of its SMMGL card is 12, and this SID contains AMMG 1 and AMMG 2)

   b) The TO_MMG points to a SID = 34 (the SID of the SMMGL card is 34, and this SID contains AMMG 3 and AMMG 4)

   Then, AMMG 1, if switched, will become AMMG 3, and AMMG 2, if switched, will become AMMG 4.

2. **History Variables.** In the switch example above, if AMMG 1 in FR_MMG has n_1 history variables and its corresponding ALE group AMMG 3 in TO_MMG has n_3 history variables, the number of history variables copied from AMMG 1 to AMMG 3 is $\min(n_1, n_3)$.

*ALE_FRAGMENTATION

Purpose: When a material fails, this card is used to switch the failed material to vacuum. When used with FRAGTYP = 2, it can be used to model material fragmentation.

Card 1	1	2	3	4	5	6	7	8
Variable	FR_MMG	TO_MMG	FRAGTYP					
Type	I	I	I					
Default	none	none	none					

VARIABLE	DESCRIPTION
FR_MMG	This is the AMMGID of the failed material.
TO_MMG	This is the AMMGID of the vacuum to which the failed material is being switched.
FRAGTYP	Flag defining whether the failed material is completely or partially switched to vacuum. EQ.1: Fully switch; all failed material is switched to vacuum (see Remark 1). EQ.2: Partially switch; only the volume expansion from the last time step is switched to vacuum (see Remark 2).

Remarks:

The Lagrange element contains only one material. Once the failure criterion is met in a Lagrange element, the whole element is marked as "failed" and either deleted or kept from further element force calculation.

However, for multi-material ALE elements, such an approach is not practical as these elements are occupied by multiple materials. Failure, therefore, cannot be adequately modeled at the element level. Instead we convert the failed material inside an ALE element to vacuum. The effect is similar to element deletion in Lagrange simulations. The failed material, once switched to vacuum, is excluded from any future element force calculation.

1. **Switch to Vacuum, (FRAGTYPE = 1).** By default, multi-material elements switch failed materials to vacuum. This switch involves assigning the full

volume fraction of the failed material, say AMMG 1, in an element to vacuum, say AMMG 2.

FRAGTYP = 1 is equivalent to the default treatment. However, with this card the vacuum AMMG can be explicitly specified. In the case that more than one vacuum AMMG exist, it is strongly recommended to use the FRAGTYP = 1 approach to eliminate ambiguity. It is also helpful during post-processing since it is possible to see the material interface of the switched material by assigning a dedicated vacuum AMMG to the switched material.

2. **Fragmentation, (FRAGTYPE = 2).** FRAGYP = 2 models material fragmentation. Note that the FRAGTYP = 1 approach leads to loss of mass and, consequently, dissipates both momentum and energy. With FRAGTYP = 2, instead of converting the full volume of the failed material to vacuum, LS-DYNA only converts the material expansion to vacuum. This approach conserves mass and, therefore, momentum and energy.

   To illustrate how this fragmentation model works, consider a tension failure example. At the time step when the material fails, LS-DYNA calculates the material expansion in the current step and converts this volume to vacuum. The stresses and other history variables are left unchanged, so that in the next time step it will again fail. The expansion in the next time step will be also converted to vacuum. This process continues until maybe at a later time the gap stops growing or even starts to close due to compression.

Example:

Consider a simple bar extension example:

FR_MMG: H5 = AMMG1 = Metal bar
H6 = AMMG2 = Ambient air
TO_MMG: H7 = AMMG3 = Dummy vacuum part

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_FRAGMENTATION
$    FR_MMG    TO_MMG   FRAGTYP
         1         3         2
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_FSI_PROJECTION

Purpose: This card provides a coupling method for simulating the interaction between a Lagrangian material set (structure) and ALE material set (fluid). The nearest ALE nodes are projected onto the Lagrangian structure surface at each time step. This method does not conserve energy, as mass and momentum are transferred using the constraint based approach.

Card 1	1	2	3	4	5	6	7	8
Variable	LAGSID	ALESID	LSIDTYP	ASIDTYP	SMMGID	ICORREC	INORM	
Type	I	I	I	I	I	I	I	
Default	0	0	0	0	0	0	0	

Card 2	1	2	3	4	5	6	7	8
Variable	BIRTH	DEATH						
Type	F	F						
Default	0.0	10^{10}						

VARIABLE	DESCRIPTION
LAGSID	A set ID defining the Lagrangian part(s) for this coupling (structures).
ALESID	A set ID defining the ALE part(s) for this coupling (fluids).
LSIDTYP	Lagrangian set ID type: EQ.0: Part set ID (PSID), EQ.1: Part ID (PID).
ASIDTYP	ALE set ID type: EQ.0: Part set ID (PSID), EQ.1: Part ID (PID).

VARIABLE	DESCRIPTION
SMMGID	A set ID referring to a group of one or more ALE-Multi-Material-Group (AMMG) IDs which represents the ALE materials interacting with the Lagrangian structure. This SMMGID is a set ID defined by *SET_MULTI-MATERIAL_GROUP_LIST.
ICORREC	Advection error correction method (see Remark 1): EQ.1: ALE mass is conserved. Leaked mass is moved, EQ.2: ALE mass is almost conserved, EQ.3: No correction performed (default). ALE mass is conserved. Some leakage may occur. This may be the best solution.
INORM	Type of coupling: EQ.0: Couple in all directions, EQ.1: Couple in compression and tension (free sliding), EQ.2: Couple in compression only (free sliding). This choice requires ICORREC = 3.
BIRTH	Start time for coupling.
DEATH	End time for coupling.

Remarks:

1. **Advection Errors.** As the ALE nodes are projected onto the closest Lagrangian surface, there may be some advection errors introduced. These errors may result in a small element mass fraction being present on the "wrong" side of the coupled Lagrangian surface. There are 3 possible scenarios:

   a) Mass on the wrong side of the Lagrangian structure may be moved to the right side. This may cause P oscillations. No leakage will occur.

   b) Mass on the wrong side is deleted. Mass on the right side is scaled up to compensate for the lost mass. No leakage will occur.

   c) Mass on the wrong side is allowed (no correction performed). Some leakage may occur. This may be the most robust and simplest approach.

Example:

Model Summary:

H1 = AMMG1 = background air mesh
H2 = AMMG1 = background air mesh
S3 = cylinder containing AMMG2
S4 = dummy target cylinder for impact

The gas inside S3 is AMMG2. S3 is given an initial velocity and it will impact S4.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_MULTI-MATERIAL_GROUP
         1         1
         2         1
*SET_MULTI-MATERIAL_GROUP_LIST
        22
         2
*ALE_FSI_PROJECTION
$   LAGSID    ALESID   LSIDTYP   ASIDTYP    SMMGID   ICORREC     INORM
         3         1         1         1        22         3         2
$    BIRTH     DEATH
       0.0      20.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_FSI_SWITCH_MMG_{OPTION}

Purpose: This card is used to allow the switching of an ALE multi-material-group ID (AMMGID) of a fluid as that fluid passes across a monitoring surface. This monitoring surface may be a Lagrangian shell structure or a segment set. It does not have to be included in the Lagrangian structure set of the coupling card: *CONSTRAINED_LAGRANGE_IN_SOLID. However, at least one coupling card must be present in the model.

Available options include:

<BLANK>

ID

TITLE

An ID number (up to 8 digits) may be defined for this switch command in the first 10-character space. A title for the card may be input between the 11th and 80th characters on the title-ID line. The optional title line precedes all other cards for this command.

The user can explicitly define a title for this coupling.

ID Card. Additional card for the Title or ID keyword options.

Card ID	1	2	3	4	5	6	7	8
Variable	ID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	NQUAD	XOFF	BTIME	DTIME	NFREQ	NFOLD
Type	I	I	I	F	F	F	I	I
Default	none	0	1	0.0	0.0	1.0E20	1	0

Card 2	1	2	3	4	5	6	7	8
Variable	FR_MMG	TO_MMG	XLEN					
Type	I	I	F					
Default	none	none	0.0					

VARIABLE	DESCRIPTION
SID	A set ID defining a monitoring surface over which an ALE fluid flows across, and its ALE multi-material-group-ID (AMMGID) is switched. The monitoring surface may be a Lagrangian shell structure or a segment set. This surface, if Lagrangian, does not have to be included in the coupling definition (see Remark 4).
STYPE	Set ID type of the above SID: EQ.0: Part set ID (PSID) (default). EQ.1: Part ID (PID). EQ.2: Segment set ID (SGSID).
NQUAD	The number of flow-sensor points to be distributed over each monitoring surface/segment. There should be enough sensor points distributed to monitor the flow in each ALE element intersected by this monitoring surface (default = 1; see Remark 3).
XOFF	An offset distance away from the monitoring surface, beyond which the AMMGID switching occurs. The direction of XOFF is defined by the normal vector of the monitoring segment. This offset distance, in general, should be at least 2 ALE element widths away from, and beyond, the monitoring interface (default = 0.0).
BTIME	Start time for the AMMGID switch to be activated (default = 0.0).
DTIME	Ending time for the AMMGID switch (default = 1.0E20).
NFREQ	Number of computational cycles between ALE switch check (default = 1).
NFOLD	Flag for checking folding logic: EQ.0: Off EQ.1: On. LS-DYNA will check if the monitoring segment is in

VARIABLE	DESCRIPTION
	the fold which is applicable to airbags. If the monitoring segment is still located within a folded (shell) region, then no switching is allowed yet until it has unfolded.
FR_MMG	This is the AMMG SID before the switch. The AMMG SID corresponds to the SID defined under the *SET_MULTI-MATERIAL_GROUP_LIST (SMMGL) card. This SID points to one or more AMMGs. See Remark 1.
TO_MMG	This is the AMMG SID after the switch. The AMMG SID corresponds to the SID defined under the *SET_MULTI-MATERIAL_GROUP_LIST card. This SID points to one or more AMMGs. See Remark 1.
XLEN	This is an absolute distance for distributing the flow sensor points over the ALE elements. To make sure that at least 1 sensor point, defined on each Lagrangian segment, is present in each ALE element to track the flow of an AMMG, XLEN may be estimated as roughly half the length of the smallest ALE element in the mesh. See Remark 3.

Remarks:

1. **AMMG SIDs.** There is a correspondence between the FR_MMG and TO_MMG. Consider an example where:

   a) The FR_MMG SID points to a SID = 12 (the SID of its SMMGL card is 12, and this SID contains AMMG 1 and AMMG 2)

   b) The TO_MMG points to a SID = 34 (the SID of the SMMGL card is 34, and this SID contains AMMG 3 and AMMG 4)

   Then, AMMG 1, if switched, will become AMMG 3, and AMMG 2, if switched, will become AMMG 4.

2. **Database files.** The ID option must be activated if the parameter SWID is used in the *DATABAS_FSI card. Then the accumulated mass of an AMMG that goes through a tracking surface, and being switched, will be reported via the parameter "mout" in the dbfsi ASCII output file (or equivalently the "POROSITY" parameter inside LS-PrePost ASCII plotting option).

3. **Sensor point interval.** When both NQUAD and XLEN are defined, whichever gives smaller sensor-point interval distance will be used. XLEN may give better control as in the case of a null shell acting as the monitoring surface. As this null

shell is stretched, NQUAD distribution of sensor points may not be adequate, but XLEN would be.

4. **Coupling cards.** The monitoring surface does not have to be included in the Lagrangian structure set of the coupling card. However, at least one coupling card must be present in the model. The monitoring segment set can be made up of Lagrangian or ALE nodes.

Example:

Consider a simple airbag model with 3 part IDs:

H25 (AMMG1) = Inflator gas injected into the airbag.
H24 (AMMG2) = Air outside the airbag (background mesh).
H26 (AMMG3) = Dummy AMMG of inflator gas after it passes through a vent hole.
S9 = A Lagrangian shell part representing a vent hole.
S1 = A Lagrangian shell part representing the top half of an airbag.
S2 = A Lagrangian shell part representing the bottom half of an airbag.

The inflator gas inside the airbag is distinguished from the inflator gas that has passed through the monitoring surface (vent hole) to the outside of the airbag by assigning different ALE multi-material group set ID to each. The dummy fluid part (H26) should have the same material and EOS model IDs as the before-switched fluid (H25).

TO_MMG = 125
⇒ AMMGID (before switch) = *SET_MULTI-MATERIAL_GROUP_LIST(125) = 1
⇒ PART = PART(AMMGID1) = H25

FR_MMG = 126
⇒ AMMGID (before switch) = *SET_MULTI-MATERIAL_GROUP_LIST(126) = 3
⇒ PART = PART(AMMGID3) = H26

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_MULTI-MATERIAL_GROUP
        25         1
        24         1
        26         1
*DATABASE_FSI
$     TOUT                                  [STYPE: 0=PSID ; 1=PID ; 2=SGSID]
    0.1000
$ DBFSI_ID       SID     STYPE  AMMGSWID  LDCONVID
         1         1         1
         2         2         1
         3         9         1     90000
*SET_MULTI-MATERIAL_GROUP_LIST
       125
         1
*SET_MULTI-MATERIAL_GROUP_LIST
       126
```

```
         3
*ALE_FSI_SWITCH_MMG_ID
     90000
$       SID   SIDTYPE     NQUAD      XOFF     BTIME     DTIME     NFREQ      FOLD
         9         1         3     -20.0       5.0       0.0         1         1
$    Fr_MMG    To_MMG     XCLEN
       125       126        5.
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

Note:

1. The *DATABASE_FSI card tracks 3 surface entities: (a) top half of an airbag, (b) bottom half of an airbag, and (c) the vent hole monitoring surface where the AMMGID of the inflator gas is switched.

2. The amount of mass passing through the vent hole during the switch is output to a parameter called “pleak” in a dbfsi ASCII file. See *DATABASE_FSI.

3. The *ALE_FSI_SWITCH_MMG_ID card track any flow across S9 and switch the AMMGSID from 125 (AMMG 1) to 126 (AMMG 3).

*ALE_FSI_TO_LOAD_NODE

Purpose: Output the ALE coupling forces from *CONSTRAINED_LAGRANGE_IN_SOLID for CTYPE = 4 in keyword format, so they may be applied directly in another run.

Card 1	1	2	3	4	5	6	7	8
Variable	DT	NSID	IOPT					
Type	I	I	I					
Default	none	none	0					

Remaining card is optional.†

Path Card. Optional card for the directory path (see Remark 4).

Card 2	1	2	3	4	5	6	7	8
Variable	PATH							
Type	A80							
Default	optional							

VARIABLE	DESCRIPTION
DT	Output intervals
NSID	Node Set ID. See *SET_NODE.
IOPT	Options to map the coupling data between 2 runs: EQ.0: The keyword alefsiloadnode.k (see Remark 1) is created at the end of the run by LS-DYNA. EQ.1: A database of coupling forces is dumped without the conversion to a keyword file at the end of the run (see Remark 2). The database can be treated by a program (alefsiloadnode.exe) to write alefsiloadnode.k. EQ.2: The database of coupling forces created by IOPT = 1 (see Remark 2) is read back. The structure meshes should be

VARIABLE	DESCRIPTION
	identical. The forces are directly applied to the nodes without using *LOAD_NODE. The parameter DT is not read, but NSID is read.
	EQ.3: A database of coupling accelerations is dumped at the end of the run (see Remark 3).
	EQ.4: The database of coupling accelerations created by IOPT = 3 (see Remark 3) is read back. The structure meshes can be different. The accelerations are interpolated at the nodes provided by NSID. The parameters DT and NSID are read.
PATH	Path to the directory where the databases are created.

Remarks:

1. **Keyword Output File.** The name of the output keyword file is alefsiloadnode.k. For each node, this file contains three *LOAD_NODE for each global direction and three *DEFINE_CURVE for the coupling force histories.

2. **Database for IOPT = 1 and 2.** The name of the database is alefsi2ldnd.tmp (or alefsi2ldnd.tmp00… in MPP). It should be in the directory of the second run for IOPT = 2. The database lists the coupling forces by node. The structure meshes (and their MPP decomposition) for the IOPT = 1 and IOPT = 2 runs should be the same.

3. **Databases for IOPT = 3 and 4.** The names of the databases are alefsi2ldnd.tmp (or alefsi2ldnd.tmp00… in MPP) and alefsi2lndndx.tmp. They should be in the directory of the 2nd run for IOPT = 4. The file alefsi2ldnd.tmp lists the coupling accelerations by node (coupling acceleration = coupling force / nodal mass). The file alefsi2lndndx.tmp lists the initial nodal coordinates and coupling segment connectivities (see *CONSTRAINED_LAGRANGE_IN_SOLID for the coupling segment). The structure meshes for the IOPT = 3 and IOPT = 4 runs can be different. The IOPT = 3 initial geometry stored in alefsi2lndndx.tmp is used to interpolate the coupling accelerations (saved in alefsi2ldnd.tmp) at the nodes provided by NSID. For the interpolation to work, these nodes should be within the IOPT = 3 coupling segment initial locations.

4. **Directory path.** The filenames of the databases can be prefixed by the variable PATH to control in which directory the databases should be created. Since the directory path and file name are concatenated, PATH should end with "/" (or "\" in Windows).

*ALE_INJECTION

Purpose: Select nodes and elements in a user defined volume to prescribe the velocities of the selected nodes and set the materials in the selected elements with material(s) from a multi-material group set.

Card 1	1	2	3	4	5	6	7	8
Variable	MMGSET	SEGSET	GLOBAL	LCE	LCRVL			
Type	I	I	I	I	I			
Default	none	none	0	0	0			

Remaining cards are optional.†

Velocity Card. Optional card for the nodal motion (see Remark 1).

Card 2	1	2	3	4	5	6	7	8
Variable	LCVT	VECT	LCVR	VECR	BOXV	XG	YG	ZG
Type	I	I	I	I	I	F	F	F
Default	0	0	0	0	0	0.0	0.0	0.0

Surface Card. Optional card for defining the geometry to be filled by MMGSET (see Remark 1)

Card 3	1	2	3	4	5	6	7	8
Variable	SURFCT	NDIV	XL	YL	ZD	ZU	XC	YC
Type	I	I	F	F	F	F	F	F
Default	0	3	0.0	0.0	0.0	0.0	0.0	0.0

VARIABLE	DESCRIPTION
MMGSET	Multi-Material Set ID (see *SET_MULTI-MATERIAL_GROUP_-LIST).
SEGSET	Segment set ID (see *SET_SEGMENT). A local coordinate system is created for each segment. See Remark 2.
GLOBAL	Three digit flag to control how to select the elements, how to prescribe the velocities, and how to define the geometrical parameters of Cards 2 and 3 (including BOXV): EQ.__0: Geometrical parameters are local to the segments of SEGSET EQ.__1: Geometrical parameters are natural to SEGSET segments (see Remark 3 and Figure 4-1) EQ._0_: Velocities are applied in local coordinate systems attached to each segment of SEGSET EQ._1_: Velocities are applied in the global coordinate system EQ.0__: Select the elements and nodes in the local volume around each segment of SEGSET EQ.1__: Select the elements in the global volume formed by all the segments of SEGSET. The nodes are selected in the local volume around each segment of SEGSET. EQ.2__: Select the elements and nodes in the global volume formed by all the segments of SEGSET. Velocities are applied in the global coordinate system.
LCE	Curve ID for the internal energy (see Remark 6): GT.0: Load curve ID; see *DEFINE_CURVE. See Remark 2. LT.0: -LCE is the function ID for the internal energy which depends on 26 arguments: time, number of cycles, and nodal coordinates of the 8 nodes for the ALE element. See *DEFINE_FUNCTION. See Remark 5.
LCRVL	Curve ID for the relative volume (see Remark 6): GT.0: Load curve ID; see *DEFINE_CURVE. See Remark 2. LT.0: -LCRVL is the function ID for the relative volume which depends on 26 arguments: time, number of cycles, and nodal coordinates of the 8 nodes for the ALE element. See *DEFINE_FUNCTION. See Remark 5.

VARIABLE	DESCRIPTION
LCVT	Curve ID for the translational velocity: GT.0: Load curve ID; see *DEFINE_CURVE. LT.0: -LCVT is the function ID for the translational velocity which depends on 5 arguments: time, number of cycles, and nodal coordinates. See *DEFINE_FUNCTION. See Remark 5.
VECT	Vector to orient the translation. See *DEFINE_VECTOR.
LCVR	Curve ID for the rotational velocity: GT.0: Load curve ID; see *DEFINE_CURVE. LT.0: -LCVR is the function ID for the rotational velocity which depends on 5 arguments: time, number of cycles, and nodal coordinates. See *DEFINE_FUNCTION. See Remark 5.
VECR	Vector to orient the rotational axis (see *DEFINE_VECTOR).
BOXV	Box (see *DEFINE_BOX) defining the region where the velocities are applied (see Remark 7).
XG, YG, ZG	Position of the rotation center (see Remark 8).
SURFCT	Flag to define the surface, inside which the nodes and elements are selected: LT.0: -SURFCT is the function ID (see *DEFINE_FUNCTION) for the rotational velocity with 17 arguments: time, number of cycles, ALE element center coordinates, segment nodal coordinates. EQ.0: Ellipsoid: $\frac{x^2}{XL^2}+\frac{y^2}{YL^2}+\frac{z^2}{ZL^2}=1$ where ZL = −ZD if $z<0$ and ZL = ZU if $z>0$ (see Remark 9 and Figure 4-3) EQ.1: Ellipse-based cylinder: $\frac{x^2}{XL^2}+\frac{y^2}{YL^2}=1$ where −ZD < z < ZU (see Remark 10 and Figure 4-4)

VARIABLE	DESCRIPTION
	EQ.2: Truncated ellipse-based cone: $\frac{x^2}{X^2}+\frac{y^2}{Y^2}=1$ where $0<z<\text{ZD}$, $X=(1-z/\text{ZU})\text{XL}$, and $Y=(1-z/\text{ZU})\text{YL}$. Also, ZD < ZU, so ZU is the apex z-coordinate of the cone (see Figure 4-5).
	EQ.3: Drop geometry meaning a cone for $-\text{ZD}<z<0$ and half an ellipsoid for $0<z<\text{ZU}$ (see Remark 11 and Figure 4-6)
	EQ.4: Box with side lengths $-\text{XL}<x<\text{XL}$, $-\text{YL}<y<\text{YL}$, and $-\text{ZD}<z<\text{ZU}$ (see Figure 4-7)
	EQ.5: Segment based cylinder (see Remark 12 and Figure 4-8)
NDIV	Number of divisions of an element cut by the surface SURFCT to compute the volume fractions (see Remark 13 and Figure 4-2).
XL	Length of the geometry SURFCT in the local x-direction
YL	Length of the geometry SURFCT in the local y-direction
ZD	Length for the geometry SURFCT in the local z-direction for $z<0$, except for SURFCT = 2 where $z>0$. ZD can be input as a negative or positive value.
ZU	Length for the geometry SURFCT in the local z-direction for $z>0$
XC	x-coordinate in the segment of the local coordinate center (see Remark 14).
YC	y-coordinate in the segment of the local coordinate center (see Remark 14).

Remarks:

1. **Card 1 Definition Only.** If only Card 1 is defined, the velocities are not prescribed. The surface to select the elements to apply the MMGSET is a sphere inscribed in the segment.

2. **Segment Local Coordinate System.** A local coordinate system is created at the center of each segment (see Figure 4-1). The normal is the local z-axis. The local x-axis is parallel to the vector that is the average of 2 vectors: the vector from Node 1 to Node 2 and the vector from Node 4 to Node 3 for a quadrangle. For a triangle, the local x-axis is parallel to the vector from Node 1 to Node 2.

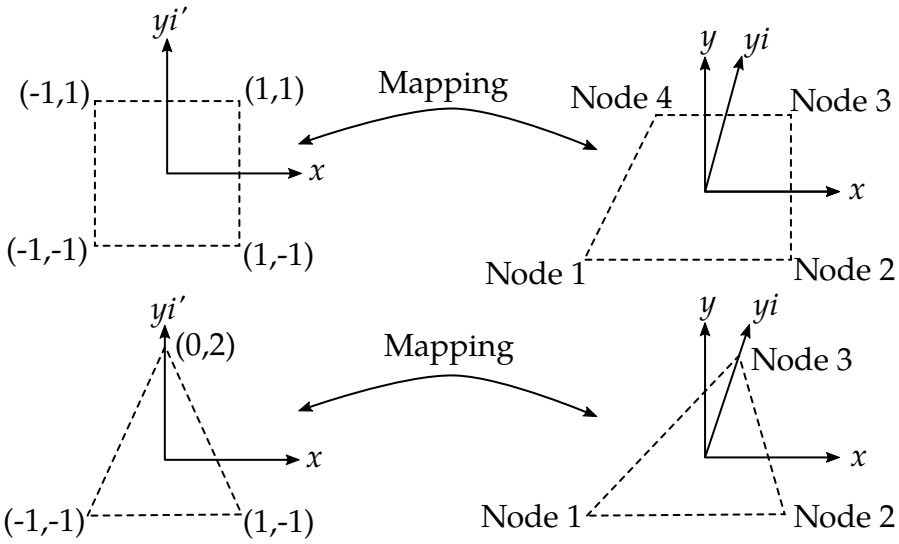

Figure 4-1. Isoparametric mapping if GLOBAL = 1

The local y-axis is the cross product of the local z- and x-axis. A 4th local axis called yi-axis can be considered for the isoparametric mapping (see Remark 3) if the segment is asymmetrical with the y-axis. For a quadrangle, the yi-axis is parallel to the vector that is an average of the 2 other sides: the vector from Node 1 to Node 4 and the vector from Node 2 to Node 3. For a triangle, the local yi-axis is parallel to the vector that points from the midpoint of the edge between Nodes 1 and 2 to Node 3.

3. **Isoparametric Mapping.** If GLOBAL = 1, the geometrical parameters, namely BOXV, XG, YG, ZG, XL, YL, ZD, ZU, XC, and YC, are natural to each segment in SEGSET. Therefore, inside the quadrangle segment, the coordinates XG, YG, ZG, XC, and YC are isoparametric, so they vary from -1 to 1, with 0 at the center. For the triangle, Node 1 is at $(-1,-1)$, Node 2 is at $(1,-1)$, and Node 3 is at $(0,2)$. The matching coordinates in the segment are in the segment-centered coordinate system (x,yi,z) (see Remarks 2 and 14). The coordinates in BOXV and lengths XL, YL, ZD, ZU are multiples of the characteristic lengths in each direction of the segment-centered orthonormal coordinate system (x,y,z) (see Remarks 2 and 14). For quadrangles, the characteristic length in the x-direction is an average of the lengths of the side from Node 1 to Node 2 and the side from Node 3 to Node 4. Similarly, the characteristic length in the y-direction is an average of the of the lengths of the side from Node 1 to Node 4 and the side from Node 2 to Node 3. For triangles, the characteristic length in the x-direction is the length of the side between Nodes 1 and 2; the characteristic length in the y-direction is the length of the median that joins the side length of Nodes 1 and 2 to the vertex of Node 3. For the z-direction, the characteristic length is an average of the characteristic x- and y-lengths.

4. **Load Curve Birth and Death Times.** The smallest and largest times in the curves are the birth and death times. If these time frames are equal to (or slightly

larger than) the time step, the related velocities can be initialized during the first cycle. If the first abscissa is a negative time with no ordinate, the curve is considered periodic and the negative time is the period.

5. **Function Birth and Death Times.** Birth and death times for functions can be considered. The functions are called twice:

   a) the first time with a negative time to determine if the function will be applied. The returned value is positive if the function is to be applied. If the returned value is negative, then the function is not applied.

   b) the second time to evaluate the function if the value returned from the first function call is positive.

6. **Internal Energy and Relative Volume Curves.** The internal energy and relative volume curves are equivalent to the ambient curves in *BOUNDARY_AMBIENT_EOS. So the materials in MMGSET should have *EOS to use these curves. If these curves are not defined, the initial values for the internal energies and relative volumes are applied (like for the ambients).

7. **Prescribed Velocity.** If BOXV is undefined, the velocities are applied to all the nodes inside the surface defined by SURFCT and the nodes of elements cut by this surface. If BOXV is defined, all the nodes inside the box have their velocities being prescribed including nodes outside the surface SURFCT.

8. **Rotation Center.** If the rotation center, namely XG, YG, and ZG, is undefined, the rotation is applied around an axis defined by VECR passing through the center of mass of the groups defined by MMGSET. In an ALE 2D model, this center is shifted to the y-axis.

9. **Ellipsoid Geometry.** The ellipsoid can have different lengths above ($z > 0$) and below ($z < 0$) the segment (see Figure 4-3). If only XL is defined, YL = ZD = ZU = XL, so the surface geometry is the surface of a sphere.

10. **Axisymmetric Cylinder Geometry.** If only XL is defined, YL = XL, so the surface is defined as an axisymmetric cylinder (see Figure 4-4).

11. **Drop Geometry.** For SURFCT = 3, the geometry is an ellipsoid-based cone is below the segment ($z < 0$) and half of an ellipsoid above the segment ($z > 0$). The geometry looks like a drop (or an ice cream cone. See Figure 4-6).

12. **Segment Projection Geometry.** For SURFCT = 5, a node or element center is inside this geometry if $-ZD < z < ZU$ and if the node or element center projection on the segment plane is inside this segment (see Figure 4-8).

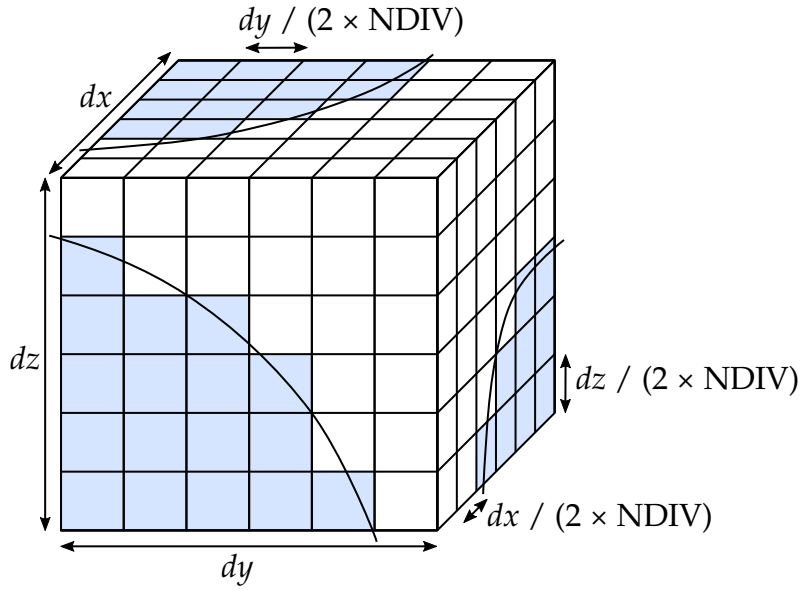

Figure 4-2. Division of an ALE element into $8 \times \text{NDIV}^3$ sub-cells to compute the volume fractions of elements cut by the surface defined by SURFCT

13. **Sub-Volumes.** Like NTRACE in *INITIAL_VOLUME_FRACTION_GEOMETRY, the division of an element into sub-volumes allows the computation of volume fractions of an element cut by the surface SURFCT. The definition of division differs from that for NTRACE in *INITIAL_VOLUME_FRACTION_GEOMETRY. For the initialization of *s*the volume fractions, the number of sub-volumes along an element edge is 2 × NTRACE + 1 for *INITIAL_VOLUME_FRACTION_GEOMETRY while it's 2 × NDIV for this keyword (see Figure 4-2).

14. **Segment Center.** The segment center is the average position of the segment nodes. By default, the local coordinate system is centered at this point. XC and YC can shift this center in the segment plane.

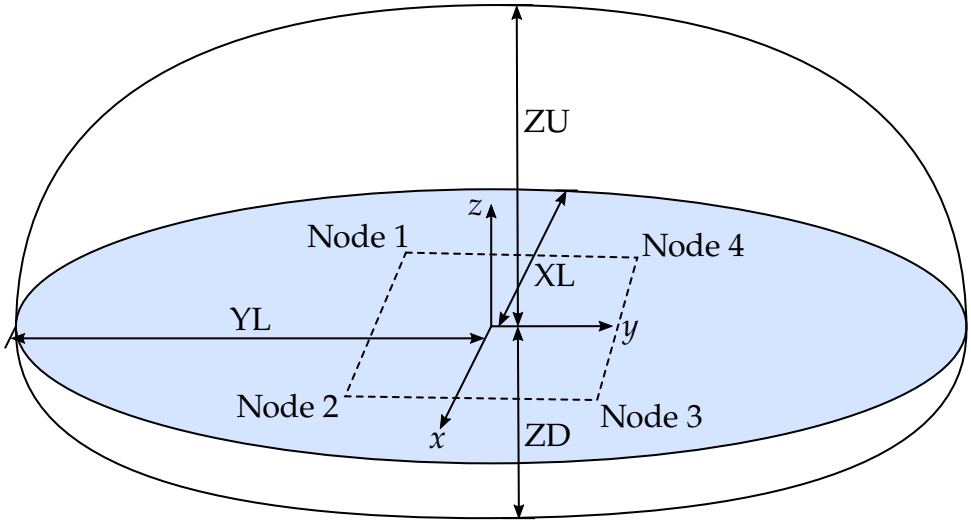

Figure 4-3. Surface for SURFCT = 0

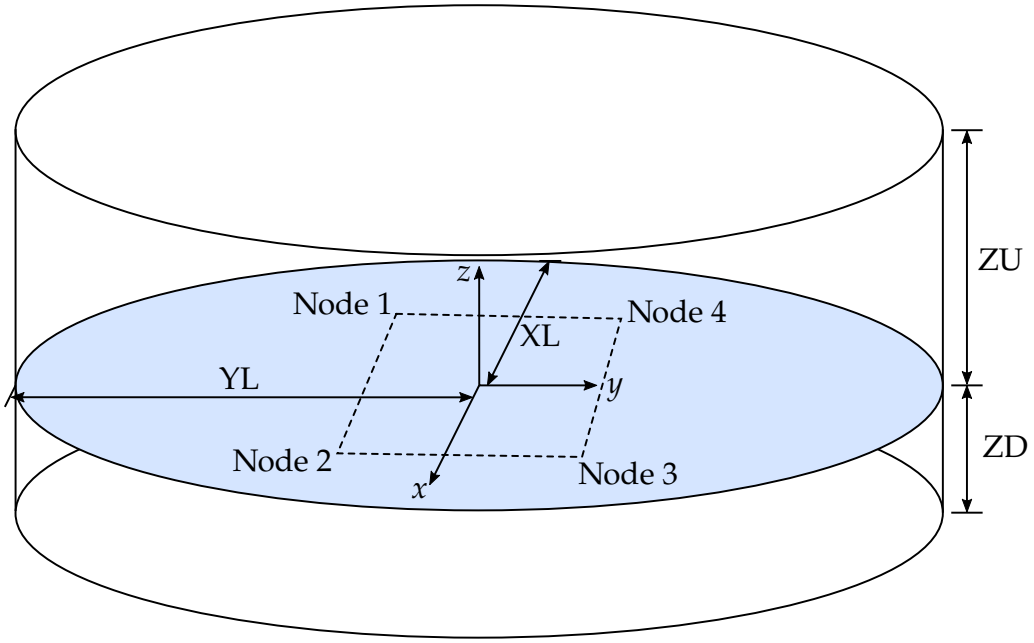

Figure 4-4. Surface for SURFCT = 1

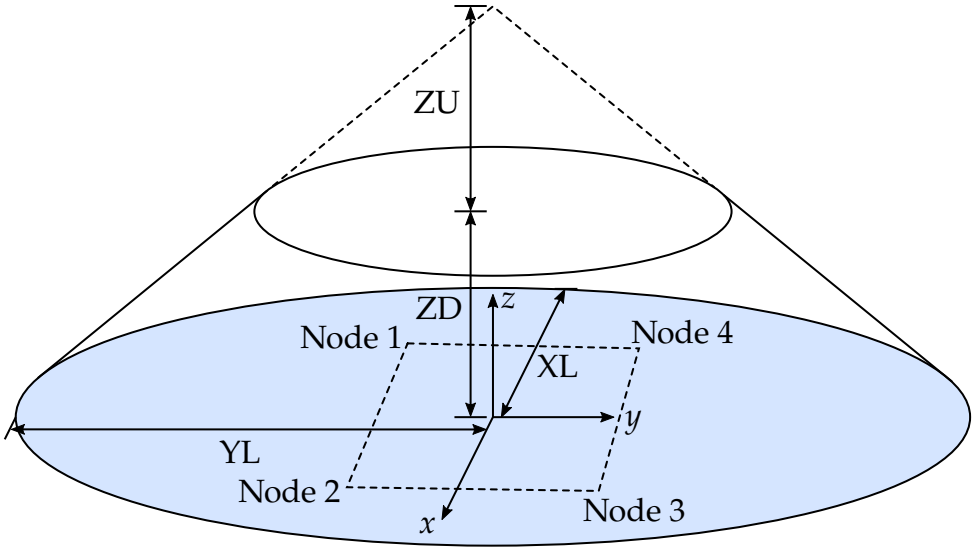

Figure 4-5. Surface for SURFCT = 2

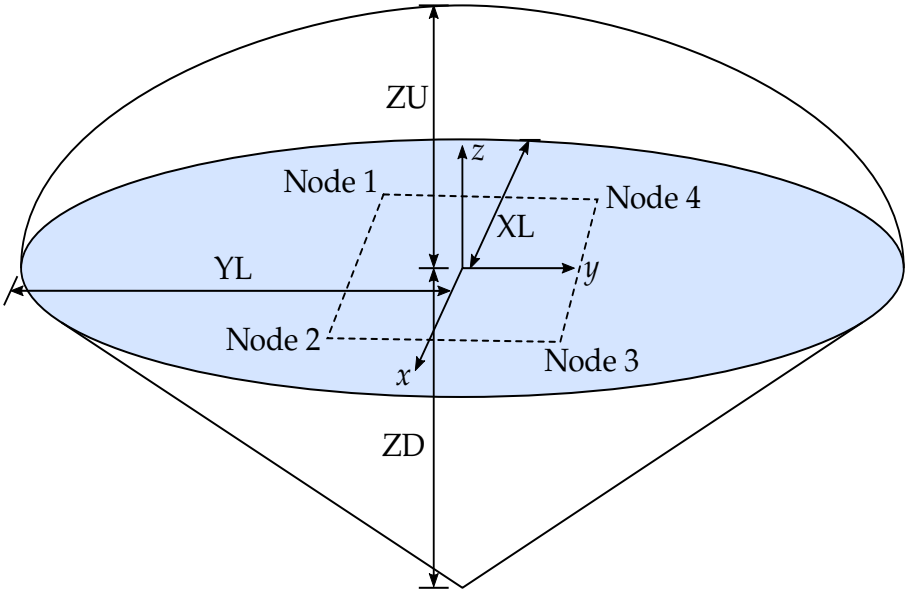

Figure 4-6. Surface for SURFCT = 3

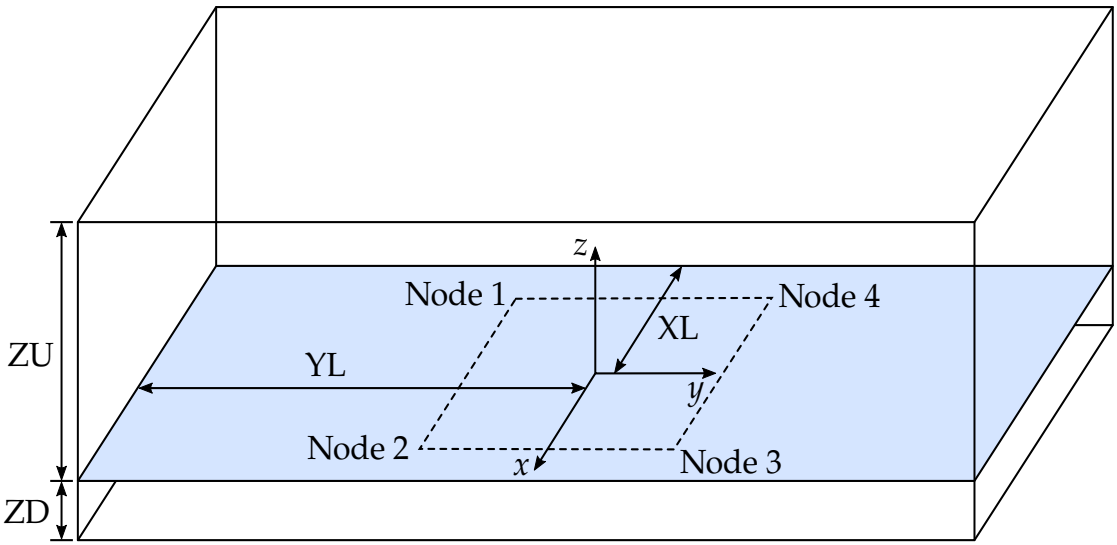

Figure 4-7. Surface for SURFCT = 4

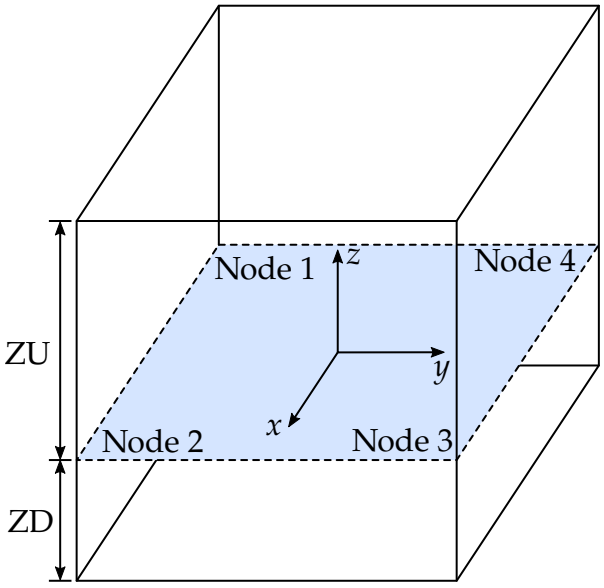

Figure 4-8. Surface for SURFCT = 5

*ALE_MAPPING

Purpose: This card maps ALE data from a previous run to a region of elements at a given time. Data are read from or written to a mapping file specified with "map=" on the command line (see Remark 1). This keyword is supported for S-ALE and ALE. To map data at the initial time to the whole ALE domain (not just a region of elements) see *INITIAL_ALE_MAPPING.

The following transitions are allowed:

1D → 2D	2D → 2D	3D → 3D
1D → 3D	2D → 3D	

Card Summary:

Card 1. This card is required.

AMMSID	RW						

Card 2. This card and all subsequent cards are optional.

NTIM	TBEG	TEND					

Card 3. This card is optional. It defines the location of the mapping for RW > 0. For RW = -1, if the subsequent cards are included, this card must be included as a blank line (all entries will be ignored).

VECID	ANGLE	XP	YP	ZP			

Card 4. This card is optional.

ID	TYPE	NVOL					

Card 4.1. Include NVOL of this card.

VOLTYP	VECID1	DW1	XL	YL	ZL	DW2	DV2

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	AMMSID	RW						
Type	I	I						
Default	none	-1						

VARIABLE	DESCRIPTION
AMMSID	Set ID of ALE multi-material groups defined in *SET_MULTI-MATERIAL_GROUP. See Remark 2.
RW	Flag defining if the keyword reads or writes in the mapping (see Remarks 3 and 4): EQ.-1: Write to the mapping file. See Remark 4. GT.0: Read from the mapping file. The magnitude of RW defines which part of the output to the mapping file is to be read in if more than one *ALE_MAPPING keywords was used to write the mapping file in the previous run. For example, if there was only one keyword used for writing the mapping file (most of the cases), set RW = 1 during the read.

"When" Card. Optional card to define the times of the mapping (see Remark 4).

Card 2	1	2	3	4	5	6	7	8
Variable	NTIM	TBEG	TEND					
Type	I	F	F					
Default	1	↓	ENDTIM					

VARIABLE	DESCRIPTION
NTIM	Number of times to write to the mapping file or data entry for the time selected to be read from the mapping file (see Remark 4):

VARIABLE	DESCRIPTION
	RW.EQ.-1: Number of times to write to the mapping file between the times TBEG and TEND. RW.GT.0: Data entry to be read if, during the previous run, a keyword *ALE_MAPPING with RW = -1 wrote several times to the mapping file. In most cases, only one time was output in which case NTIM = 1.
TBEG	RW.EQ.-1: Time to start writing to the mapping (TBEG = ENDTIM by default). See Remark 4. RW.GT.0: Time to which the read data entry indicated by NTIM will be mapped (TBEG = 0.0 by default).
TEND	For RW = -1 only, time to stop writing to the mapping file. See Remark 4.

"Where" Card. Optional card to define the location of the mapping for RW > 0 (see Remarks 5 and 6). For RW = -1, if the subsequent cards are included, this card must be included as a blank line (all entries will be ignored).

Card 3	1	2	3	4	5	6	7	8
Variable	VECID	ANGLE	XP	YP	ZP			
Type	I	F	F	F	F			
Default	0	0.0	0.0	0.0	0.0			

VARIABLE	DESCRIPTION
VECID	ID of the symmetric axis defined using *DEFINE_VECTOR. The fields XT, YT, and ZT in *DEFINE_VECTOR define the location of the origin in the previous run. See Remarks 5 and 6.
ANGLE	Angle of rotation in degrees around an axis defined by *DEFINE_VECTOR for the 3D to 3D mapping. See Remark 6.
XP	x-position of a point on a plane used by specific mappings (only for 2D plain strain to 3D mappings). See Remark 6.
YP	y-position of a point on a plane used by specific mappings (only for 2D plain strain to 3D mappings). See Remark 6.

VARIABLE	DESCRIPTION
ZP	z-position of a point on a plane used by specific mappings (only for 2D plain strain to 3D mappings). See Remark 6.

Region Card. Optional card to define which elements are targeted by the mapping (see Remark 7).

Card 4	1	2	3	4	5	6	7	8
Variable	ID	TYPE	NVOL					
Type	I	I	I					
Default	0	0	0					

VARIABLE	DESCRIPTION
ID	Part ID, part set ID, or element set ID. See Remark 7.
TYPE	Type of "ID" (see Remark 7): EQ.0: part set ID EQ.1: part ID EQ.2: element set ID
NVOL	Number of volumes in which the elements are selected for the mapping. See Remark 7.

Volume Card. Include NVOL of this card to define the volumes selecting the elements targeted by the mapping. See Remarks 7, 8, and 9.

Card 4.1	1	2	3	4	5	6	7	8
Variable	VOLTYP	VECID1	DW1	XL	YL	ZL	DW2	DV2
Type	I	I	F	F	F	F	F	F
Default	none	0	0.0	0.0	0.0	0.0	DW1	DV1

VARIABLE	DESCRIPTION
VOLTYP	Type of volume containing the selected elements for the mapping. The magnitude of VOLTYP indicates the type of volume. VOLTYP > 0 flags that the elements *inside* the volume are selected while VOLTYP < 0 indicates that the elements *outside* the volume are selected. The volume depends on geometrical lengths in a local coordinate system defined by orthonormal axis called **u**, **v** and **w** (see Remarks 7, 8, and 9). |VOLTYP|.EQ.1: Trapezoidal prism (see Figure 4-10) |VOLTYP|.EQ.2: Elliptic truncated cone (see Figure 4-11) |VOLTYP|.EQ.3: Ellipsoid (see Figure 4-12)
VECID1	ID of the local **u**-axis defined using *DEFINE_VECTOR. See Remark 8. LE.0: Axis used to define special case volumes. See Remark 9.
DW1	Length in the local **w**-axis direction. See Remarks 8 and 9.
XL	Global *x*-position of a point used to define the local **v**-axis; the point cannot be on VECID1. See Remarks 8 and 9.
YL	Global *y*-position of a point used to define the local **v**-axis; the point cannot be on VECID1. See Remarks 8 and 9.
ZL	Global *z*-position of a point used to define the local **v**-axis; the point cannot be on VECID1. See Remarks 8 and 9.
DW2	Length in the local **w**-axis direction (DW2 = DW1 by default). See Remark 8.
DV2	Length in the local **v**-axis direction (DV2 = DV1 by default). See Remark 8.

Remarks:

1. **Mapping file.** To make one mapping, only the command-line argument "map=" is necessary. If RW = -1, the mapping file will be created and ALE data histories will be written to this file. If RW > 0, the mapping file will be read and ALE data histories will be used to interpolate the ALE variables of the selected elements.

   To include several successive mappings in the file, the prompt "map1=" is necessary. If RW = -1 and the prompt "map1=" is in the command line, the ALE

data are written to the mapping file given by "map1=." However, reading the map file is the same procedure as the single mapping case using "mapping=" on the command line.

The mapping file contains the following nodal and element data:

- nodal coordinates
- nodal velocities
- part IDs
- element connectivities
- element centers
- densities
- volume fractions
- stresses
- plastic strains
- internal energies
- bulk viscosities
- relative volumes

2. **Mapping of multi-material groups.** The routines initiated by this keyword require knowing which mesh will be initialized with the mapping data and more specifically which multi-material groups will be mapped (if RW > 0) or output (for a next run if RW = -1). The field AMMSID refers to the *SET_MULTI-MATERIAL_GROUP_LIST card. That keyword defines a list of material groups in the current run. The number of entries in this list should match the number of entries from the previous run (as a reminder the data entries of multi-material groups are defined in *ALE_MULTI-MATERIAL_GROUP). For instance, if the previous model has 3 groups and the current one has 5 groups, the following is a possible mapping:

   The 1st group (previous) ⇒ the 3rd group (current),
   The 2nd group (previous) ⇒ the 5th group (current) and,
   The 3rd group (previous) ⇒ the 4th group (current).

   In this case, the *SET_MULTI-MATERIAL_GROUP_LIST card should be as follows:

```
*SET_MULTI-MATERIAL_GROUP_LIST
300
3,5,4
```

3. **Several keywords writing to the mapping file.** Several instantiations of *ALE_MAPPING in a single run with RW = -1 can write to the mapping file. The

order with which the instantiations write to the file is important for retrieving the correct data. The first definition of *ALE_MAPPING with RW = -1 in the input deck will have the first position, the second *ALE_MAPPING with RW = -1 will have the second position, and so on. For instance, if 3 *ALE_MAPPING keywords output data to the mapping file during a previous run, then setting RW = 2 in the current run will cause LS-DYNA to read the data written in the previous run from the second instantiation.

4. **Writing several times to the mapping file with a single keyword.** If NTIM > 1, *ALE_MAPPING with RW = -1 can write several times in the mapping file. When the time reaches TBEG, the data for TBEG are written to the mapping file. Then, data are output to the file NTIM-1 at $(TBEG - TEND)/(NTIM - 1)$ time intervals. If NTIM < 2, the data are only output for TBEG. The order of writing is in time order which is important for reading the data later. For instance, if the first run outputs data 4 times (NTIM = 4), then using RW = 1 and NTIM = 3 in a later run causes LS_DYNA to read the third time that data were written by the first run.

5. **Previous origin.** Several *ALE_MAPPING with RW > 0 can map the same set of data to different parts of the current mesh by changing the origin of the previous run. The location of the origin is set by the fields XT, YT, and ZT in the *DEFINE_VECTOR referenced with VECID.

6. **Orientation vector.** For a mapping file created by a previous asymmetric model, the symmetric axis orientation in the current model is specified by VECID. For a mapping file created by a 3D or 1D spherical model, the orientation of the vector VECID is computed but ignored. When 2D plain strain data are mapped in a 3D calculation, the point (XP,YP,ZP) together with the vector, VECID, define the plane in the 3D mesh where the mapping occurs.

7. **Elements for mapping.** It's possible to select which elements should be the target of a mapping (if RW > 0) or have their data output in a mapping file (if RW = -1). The elements can be selected by part or element sets. Along with this first selection, volumes can defined (if NVOL > 0) to adjust the selection. The intersection of these volumes and sets provides the final selection.

   If NVOL > 0, NVOL extra lines at the end of the keyword should be added to define volumes for the element selection. Each line defines a volume. If VOLTYP > 0, the elements inside the intersection of these volumes are selected. If VOLTYP < 0, the elements outside the intersection of these volumes are selected.

8. **Defining volumes.** Defining the three different kinds of volumes from the input fields requires a set of very similar steps. Defining VOLTYP = 1, the trapezoidal prism, will be discussed in detail to illustrate these steps. Then the differences

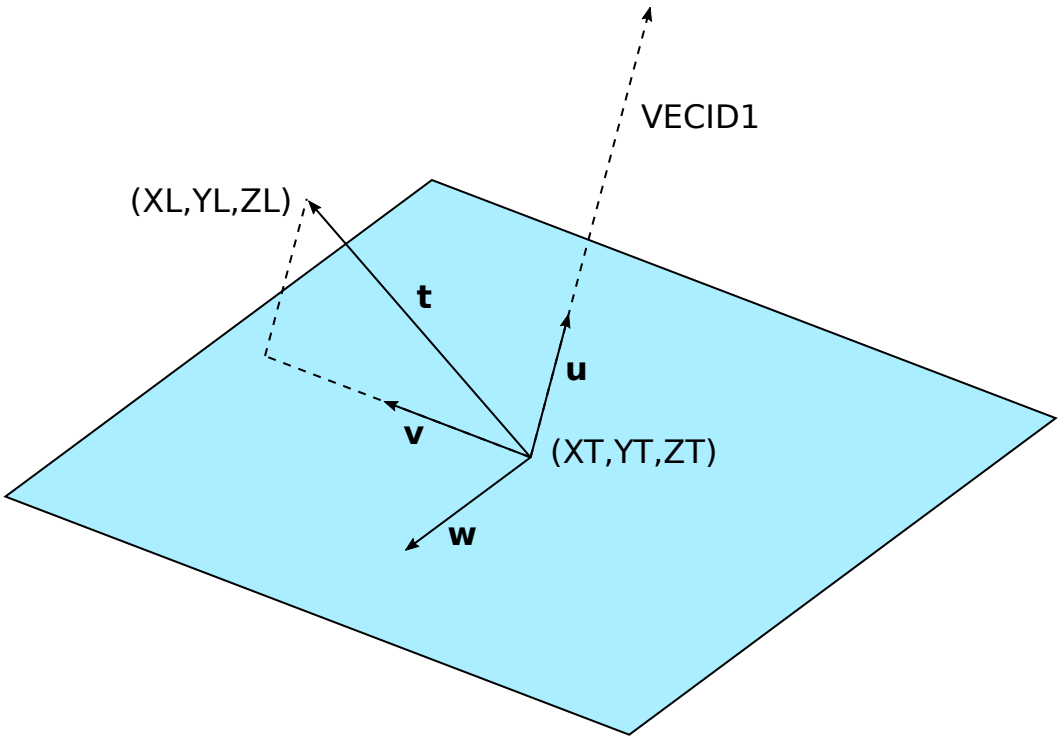

Figure 4-9. Local Coordinate System for the Defined Volumes

in how the fields are used for the other two volumes are highlighted following this discussion.

The first step required to construct the trapezoidal prism from the input fields is to define a local coordinate $(\mathbf{u}, \mathbf{v}, \mathbf{w})$.

a) *Defining the Local Coordinate System.* The local coordinate system is defined using a vector and a point, namely VECID1 and (XL,YL,ZL). The origin of the coordinate system is the tail of VECID1 which is the point (XT,YT,ZT) set in the *DEFINE_VECTOR input. See Figure 4-9.

   i) **u**-*axis.* The vector **u** is the unit direction of VECID1.

   ii) **v**-*axis.* From here the other two vectors can be constructed on the plane perpendicular to **u** using the point (XL,YL,ZL). Note that for this construction to work, (XL,YL,ZL) *cannot* be along the **u** vector. Let $\mathbf{t}$ = (XL,YL,ZL) − (XT,YT,ZT). Then, **v** is the unit vector along the projection of **t** onto the plane perpendicular to **u**.

   iii) **w**-*axis.* **w** is then the unit vector that completes the right-hand coordinate system.

The trapezoidal prism is constructed by connecting two faces defined with the local coordinate system. The two faces are both perpendicular to **u**. See Figure 4-10.

b) *First Face of the Trapezoidal Prism.* The first face is a rectangle on a plane that includes the point (XT,YT,ZT) and is perpendicular to **u**. The rectangle is centered at the point (XT,YT,ZT), and it has its sides oriented such that they align with the **v** and **w** vectors. The length of the side oriented along the **v**-axis is 2 × DV1. DV1 is the length of the projection of **t** onto the plane. The length of the side oriented along the **w**-axis is 2 × DW1, where DW1 is an input field.

c) *Second Face of the Trapezoidal Prism*. The second face of the trapezoidal is defined very similarly to the first. It is also a rectangle on a plane perpendicular to **u**, but the rectangle is centered on the point (XH,YH,ZH). This point is set in *DEFINE_VECTOR for VECID1. The sides are oriented the same way as in the first face definition. In this case the length of the side oriented along the **v**-axis is 2 × DV2 which is an input field. The length of the side oriented along the **w**-axis is 2 × DW2, where DW1 is an input field.

VOLTYP = 2, the elliptic truncated cone, is basically the same as the trapezoidal prism, except that it has two elliptic faces instead of two rectangular faces centered at the same locations as the rectangular faces. The ellipse semi-axes are aligned along **v** and **w** with lengths of DV1 and DW1 on the first face and lengths of DV2 and DW2 on the second face. DV1 is defined the same way as for the trapezoidal prism. See Figure 4-11.

VOLTYP = 3, the ellipsoid, is defined using an ellipse that is the same as the first face of the VOLTYP = 2. That ellipse is one of the symmetry planes of the ellipsoid. The ellipsoid is centered at the point (XT,YT,ZT). It has semi-axes that are the length of VECID1, DV1, and DW1 along **u**, **v**, and **w**, respectively. Both DV2 and DW2 are not used in this case. See Figure 4-12.

9. **Shortcuts for defining common types of volumes.** The keyword reader allows for certain shortcuts to define cylinders, ellipsoids with circular cross-sections, and spheres. All of these volumes can be constructed through the process described in Remark 8.

a) *Shortcut for Cylinder.* With VOLTYP = 2, a cylinder can be defined more simply with a vector and radius. To do this, let VECID1 be the axis of the cylinder, but to indicate to LS-DYNA that a shortcut is being used input the ID as a negative number. Set DW1 as the radius of the cylinder. No other input is required. The length of the cylinder is the length of VECID1. Its faces will be centered on the (XT,YT,ZT) and (XH,YH,ZH), just like an arbitrary truncated cone with ellipsoid faces.

b) *Shortcut for Ellipsoid with Circular Cross-Section.* To obtain an ellipsoid with a circular cross-section, set VOLTYP = 3 and pick a radius and the axis perpendicular to the cross-section. Set VECID1 to a the negative of the *DEFINE_VECTOR ID for the axis. Set the field DW1 as the radius. This input

will create an ellipsoid with a circular cross-section centered at the point (XT,YT,ZT). Like the normal ellipsoid definition, the remaining semi-axis will have the same length as VECID1.

c) *Shortcut for Sphere.* To obtain a sphere, set VOLTYP = 3 and VECID = 0. The only other required input are a radius which will be set in the field DW1 and a center point which will be set using (XL,YL,ZL).

10. **Comparison to *BOUNDARY_ALE_MAPPING.** Both *ALE_MAPPING and *BOUNDARY_ALE_MAPPING, can write multiple sets of data to a file at a given frequency between certain times. The big difference between these two keywords is reading the data. With *ALE_MAPPING, you select only one set of data to map from the previous run to the current run which is mapped at a time you select. With *BOUNDARY_ALE_MAPPING, all data output between t = BIRTH and t = DEATH from the previous run is mapped to the elements in the current run. The discrete histories from the previous run are interpolated.

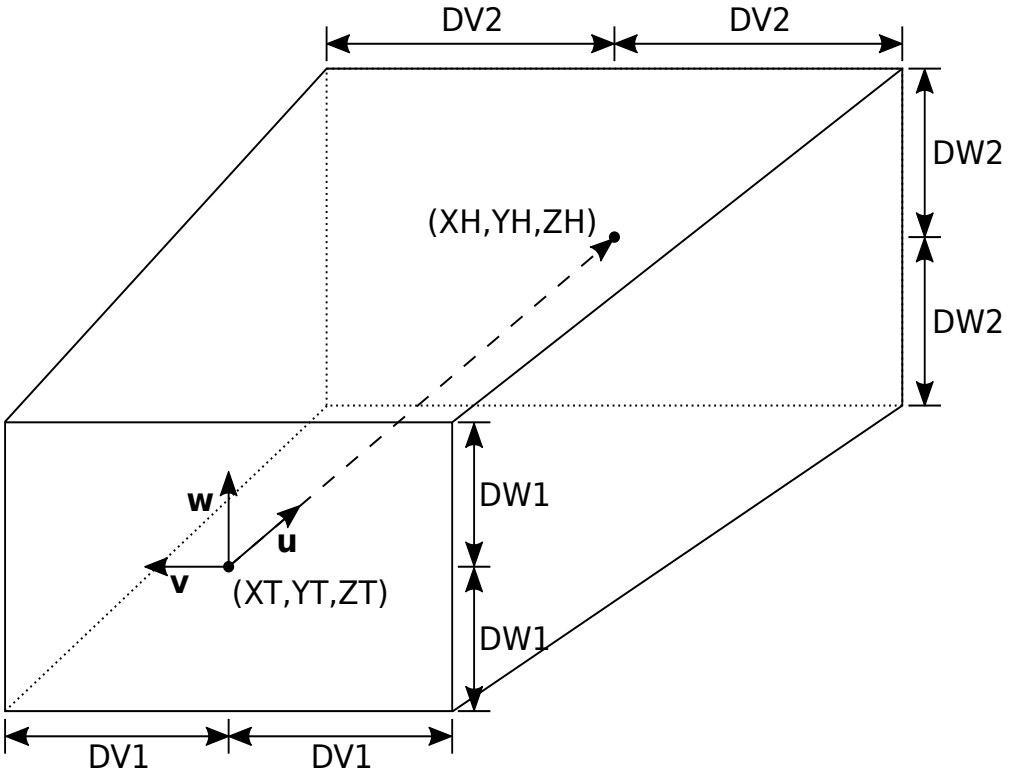

Figure 4-10. Trapezoidal Prism (VOLTYP = 1)

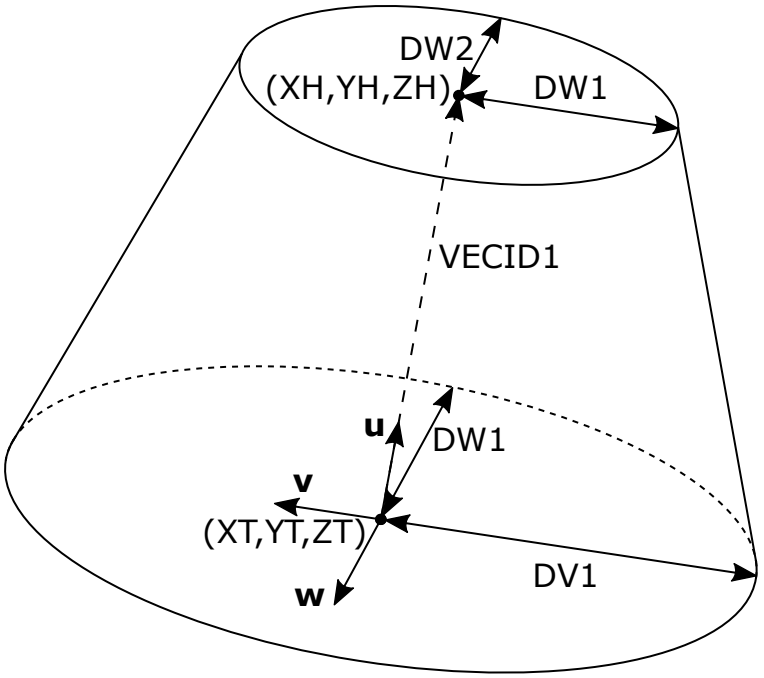

Figure 4-11. Elliptic Truncated Cone (VOLTYP = 2)

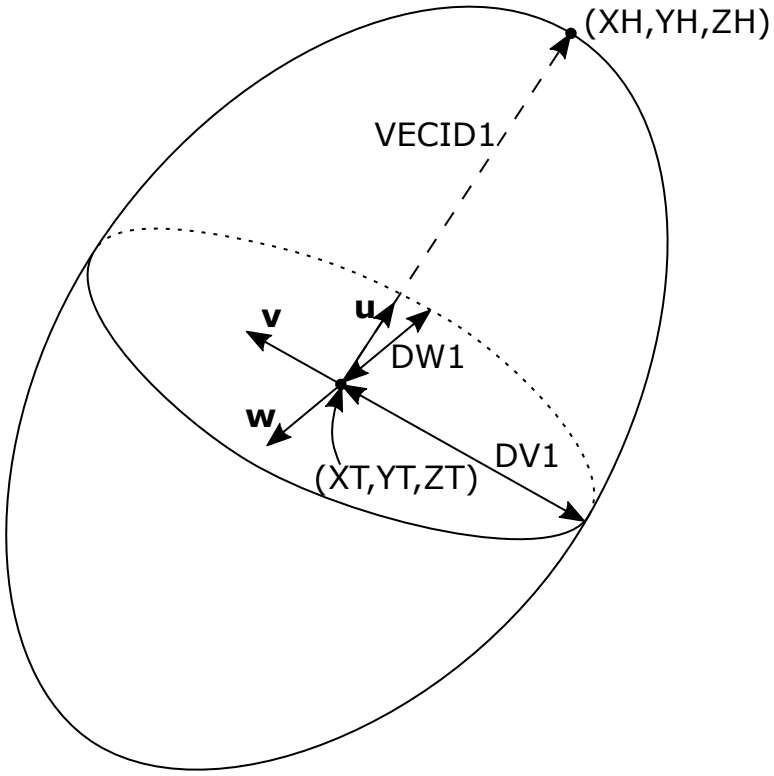

Figure 4-12. Ellipsoid (VOLTYP = 3)

*ALE_MAPPING_FROM_LAGRANGIAN

Purpose: Map data from a Lagrangian run (that is, non-ALE solid model) to ALE solids. Data are written to a mapping file called lag2ale_map (see Remark 1) that can be read back to initialize a 3D ALE mesh (see *INITIAL_ALE_MAPPING). To generate the mapping file, this keyword should be included in the input deck for the Lagrangian simulation.

Card Summary:

Card 1. This card is required to select which Lagrangian parts are mapped.

LAGPID	LAGPTYP						

Card 2. This card is required to generate the ALE mesh which Lagrangian data are mapped to.

NX	NY	NZ	NPX	NPY	NPZ	ALEPID	

Card 3. This card is optional. It controls how the volumes shared by the Lagrangian and ALE elements are computed.

METHOD	DIV						

Data Card Definitions:

Lagrangian Card. Card to select which Lagrangian parts are mapped.

Card 1	1	2	3	4	5	6	7	8
Variable	LAGPID	LAGPTYP						
Type	I	I						
Default	none	0						

VARIABLE	DESCRIPTION
LAGPID	Part or part set ID for Lagrangian parts involved in the mapping
LAGPTYP	Type of LAGPID: EQ.0: ID is a part set ID (see *SET_PART) EQ.1: ID is a part ID (see *PART)

ALE Mesh Card. Card to create the ALE mesh which Lagrangian data are mapped to.

Card 2	1	2	3	4	5	6	7	8
Variable	NX	NY	NZ	NPX	NPY	NPZ	ALEPID	
Type	I	I	I	I	I	I	I	
Default	none	none	none	0	0	0	none	

VARIABLE	DESCRIPTION
NX, NY, NZ	Number of ALE elements in each direction of the global coordinate system. These parameters create a structured box mesh. See Remark 2.
NPX, NPY, NPZ	Number of extra elements to pad the box mesh beyond its lower and upper limits in each direction of the global coordinate system. See Remark 2.
ALEPID	Part ID of the ALE mesh

Intersection Computation Card. Optional card to control how the volumes shared by the Lagrangian and ALE elements are computed (see Remark 3).

Card 3	1	2	3	4	5	6	7	8
Variable	METHOD	DIV						
Type	I	I						
Default	0	10						

VARIABLE	DESCRIPTION
METHOD	Method to compute volumes at the intersection of Lagrangian and ALE elements: EQ.0: Both METHOD = 1 and METHOD = 2 are applied by default. See Remark 3. EQ.1: The intersection volumes are exactly computed.

VARIABLE	DESCRIPTION
	EQ.2: The intersection volumes are evaluated with DIV. See Remark 3.
DIV	Division of ALE element edges to create subcells, which volumes inside Lagrangian elements are added up by METHOD = 2 to approximate the intersection volumes at the intersection between ALE and Lagrangian elements. See Remark 3.

Remarks:

1. **Output Files.** When the Lagrangian run normally terminates with this keyword in the input deck, the following files are generated:

File Name	Description
lag2ale_lagmesh.k	Lagrangian mesh selected by this keyword
lag2ale_alemesh.k	ALE mesh generated by this keyword
lag2ale_volfrac.k	Fractions of the ALE element volumes occupied by Lagrangian elements. These volume fractions are listed in this file using *INITIAL_VOLUME_FRACTION.
lag2ale_mass	Lagrangian element masses distributed across the ALE mesh
lag2ale_stress.k	Lagrangian element stresses distributed across the ALE mesh. These stresses are listed in this file with *INITIAL_STRESS_SOLID.
lag2ale_velo.k	Lagrangian velocities interpolated at ALE nodes. These velocities are listed in this file with *INITIAL_VELOCITY_NODE.
lag2ale_map	Mapping file that can be read back with the command-line argument "map = " and *INITIAL_ALE_MAPPING to initialize the run of an ALE model with a mesh similar to the one in lag2ale_alemesh.k (that is, you can opt for a different ALE mesh, but its size, position and element size should be close to lag2ale_alemesh.k for the mapping to work properly)

2. **ALE Mesh.** The ALE mesh, which Lagrangian data are mapped to, is made of identical parallelepiped hexahedral solids. The nodal connectivities and nodal coordinates are listed under *ELEMENT_SOLID and *NODE, respectively, in a file called lag2ale_alemesh.k. The structured mesh has 2 zones:

a) A core mesh is a box of NX × NY × NZ elements. The box exactly embeds the Lagrangian parts selected by LAGPID. In other words, the dimensions of all these Lagrangian parts in each direction gives the box size.

b) The core mesh can be padded with extra layers of elements. NPX, NPY, and NPZ are the number of extra elements in each global direction.

3. **Mesh Intersection.** If a Lagrangian element is superimposed with an ALE element from the mesh created by this keyword, the volume that they share can be evaluated by 2 methods:

a) *METHOD = 1.* The computation of the intersections between edges and faces yields a volume that can be calculated in most of the cases.

b) *METHOD = 2.* The intersection volume is approximated by summing the volumes of smaller DIV × DIV × DIV subcells that:

i) divide the ALE element volumes and

ii) are inside the Lagrangian element

For the default (METHOD = 0), both methods are applied. In this case, if METHOD = 1 fails, METHOD = 2 will provide the intersection volume.

*ALE_MESH_INTERFACE

Purpose: Mesh with triangular shells the material interfaces (isosurfaces of the volume fraction 0.5) of ALE groups selected by a multi-material group set. The keyword also meshes the volume of these materials with tetrahedra. The meshes are output in keyword format when the run reaches termination time or if the user applies a sense switch (Ctrl-C and then enter a SW code when prompted. See "Sense Switch Controls" in "Getting Started").

Card 1	1	2	3	4	5	6	7	8
Variable	MMGSET	NOWRT	VOLRAT	INTERP				
Type	I	I	F	I				
Default	none	0	0.0	0				

Remeshing Card. Optional card controlling remeshing of the interface (see Remark 2).

Card 2	1	2	3	4	5	6	7	8
Variable	EDGMIN	EDGMAX						
Type	F	F						
Default	0.0	0.0						

VARIABLE	DESCRIPTION
MMGSET	Multi-Material Set ID (see *SET_MULTI-MATERIAL_GROUP_-LIST). The materials (or ALE groups) in this set are selected to be meshed.
NOWRT	File output flag. NOWRT is interpreted digit-wise, NOWRT = [*PNML*]: $$\text{NOWRT} = L + M \times 10 + N \times 100 + P \times 1000$$ The 1's digit controls the output of the initial triangular mesh on the material interfaces:

VARIABLE	DESCRIPTION
	L.EQ.0: Write the initial triangular meshes for the material interfaces (see Remark 1). The mesh is output to a keyword file called alemeshmatint.k. L.EQ.1: Do not output alemeshmatint.k. The 10's digit controls the output of the remesh for the triangular mesh on the material interface: M.EQ.0: Write triangular mesh on the material interfaces after remeshing (see Remark 2) to a keyword file called aleremeshmatint.k. M.EQ.1: Do not output aleremeshmatint.k. The 100's digit controls the output of the mesh in the material volumes: N.EQ.0: Write tetrahedral mesh of the material volumes to a keyword file called alemeshmatvol.k. N.EQ.1: Do not output alemeshmatvol.k. The 1000's digit controls the output of a mapping file that can be read back with *INITIAL_LAG_MAPPING to initialize the tetrahedral mesh in alemeshmatvol.k: P.EQ.0: Write the Lagrangian mapping file called alemeshmap. P.EQ.1: Do not output alemeshmap.
VOLRAT	Mesh volume ratio beyond which the mesh is output (see Remark 3)
INTERP	Interpolation method: EQ.0: The ALE hexahedron data are interpolated at the Lagrangian tetrahedron centers. EQ.1: The intersection volumes between ALE hexahedra and Lagrangian tetrahedra are computed, and the ALE data are mapped to the Lagrangian elements with a volume-averaged method.
EDGMIN	Minimum triangle edge applied during remeshing (see Remark 2)
EDGMAX	Maximum triangle edge applied during remeshing (see Remark 2)

Remarks:

1. **Initial material interface mesh.** The isosurfaces of the volume fraction 0.5 (material interfaces) for the selected ALE groups (materials) are first meshed with triangles, whose areas depend on the ALE element face sizes and how the material interface cuts these elements. For instance, if a material interface in an ALE element is parallel to one of its faces, the area of the 2 triangles meshing the interface will be half the face area. If the material interface only cuts one corner of an ALE element, the resulting triangular surface will be one shell. In this case, if the material volume is small, the triangle meshing the material interface will be small as well. Thus, the interface mesh can have triangles of very different sizes. If the material volume needs to be meshed, a remeshing of its surface is required. If a material is in contact with the ALE mesh boundary, this contact surface will be meshed to ensure a closed volume.

2. **Remeshing of the material interface.** The material interface is remeshed based on the coordinates of the first try to mesh it. The remeshing code is the same as the one controlled by *CONTROL_REMESHING. Similar to RMIN and RMAX for that keyword, each triangle edge of the new mesh should be between a minimum EDGMIN and maximum EDGMAX. If these parameters are not provided, EDGMIN and EDGMAX are the minimum and maximum edge size of the initial mesh, respectively.

3. **Mesh volume ratio threshold.** A material can be divided into several volumes. Some of them could be so small that they would be useless or difficult to mesh. If one of these fragments has a ratio of its volume over the total material volume smaller than VOLRAT, the fragment will not be meshed.

4. **Part IDs in alemeshmatint.k.** If a material is fragmented into N volumes, each of these fragments will have a specific part ID in alemeshmatint.k. The ID for fragment i with $1 \leq i \leq N$ of the j^{th} ALE group is $\times 1000 + j$.

5. **File names.** If there are several *ALE_MESH_INTERFACE, files are output for each of these keywords. The base names for a given instance of this keyword are appended with a number that refers to the order in which that instance is included, meaning the output for the second instance, for example, would have a 2 appended.

*ALE_MULTI-MATERIAL_GROUP

Purpose: This command defines the appropriate ALE material groupings for interface reconstruction when two or more ALE Multi-Material Groups (AMMG) are present in a model. This card is required when ELFORM = 11 in the *SECTION_SOLID card or when ALEFORM = 11 in *SECTION_ALE1D or *SECTION_ALE2D. Each data line represents one ALE multi-material group (AMMG), with the first line referring to AMMGID 1, second line AMMGID 2, etc. Each AMMG represents one unique "fluid" which may undergo interaction with any Lagrangian structure in the model.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	IDTYPE						
Type	I	I						
Default	none	0						
Remarks	1							

VARIABLE	DESCRIPTION
SID	Set ID
IDTYPE	Set type: EQ.0: Part set, EQ.1: Part.

Remarks:

1. **ELFORM.** When ELFORM = 12 in the *SECTION_SOLID card (single material and void), this card should not be used. In one model, ELFORM = 12 cannot be used together with ELFORM = 11. If possible, it is recommended that ELFORM = 11 be used as it is the most robust and versatile formulation for treating multi-material ALE parts.

2. **AMMGID Assignment.** Each AMMG is automatically assigned an ID (AMMGID) and consists of one or more PART IDs. The interface of each AMMGID is reconstructed as it evolves dynamically. Each AMMGID is represented by one material contour color in LS-PrePost.

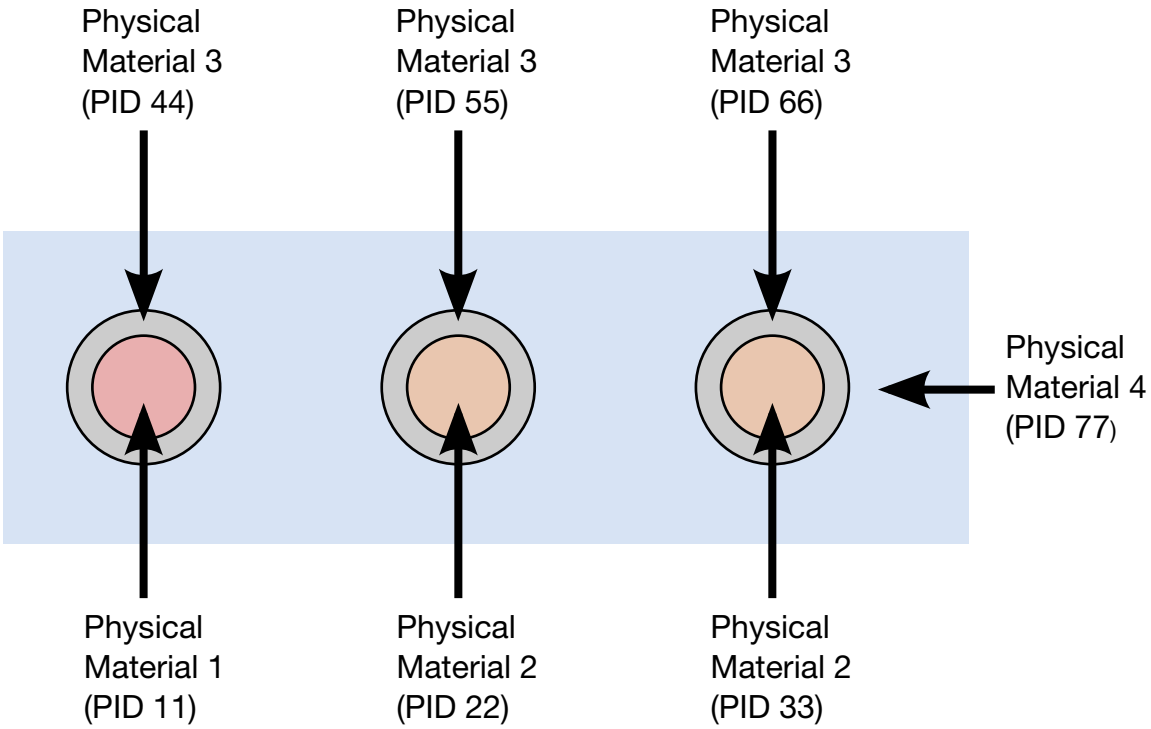

Figure 4-13. Schematic illustration of Example 1.

3. **ALE Elements and AMMGs.** The maximum number of AMMGIDs allowed has been increased to 20. However, there may be 2, at most 3, AMMGs inside an ALE element at any time. If there are more than 3 AMMGs inside any 1 ALE element, the ALE mesh needs refinement. Better accuracy is obtained with 2 AMMGs in mixed elements.

4. **Plotting AMMGIDs.** To plot these AMMGIDs in LS-PrePost:

   [FCOMP] ⇒ [MISC] ⇒ [VOLUME FRACTION OF AMMGID #] ⇒ [APPLY]

   (Note: Contour definitions maybe different for gas mixture application)

5. **Model Description.** It is very important to distinguish among the

   a) Physical materials,

   b) PART IDs, and

   c) AMMGIDs.

   A *PART may be any mesh component. In ALE formulation, it is simply a geometric entity and a time = 0.0 concept. This means a *PART may be a mesh region that can be filled with one or more AMMGIDs at time zero, via a volume filling command (*INITIAL_VOLUME_FRACTION_GEOMETRY). An AM-

MGID represents a physical material group which is treated as one material entity (represented by 1 material color contour in LS-PrePost plotting). AMMGID is used in dealing with multiple ALE or Eulerian materials. For example, it can be used to specify an ALE group in a coupling card.

Example 1:

Consider a purely Eulerian model containing 3 containers containing 2 different physical materials (fluids 1 and 2). All surrounded by the background material (maybe air). The containers are made of the same material, say, metal. Assume that these containers explode and spill the fluids. We want to track the flow and possibly mixing of the various materials. Note that all 7 parts have ELFORM = 11 in their *SECTION_SOLID cards. So we have total of 7 PIDs, but only 4 different physical materials. See Figure 4-13.

Approach 1: If we want to track only the interfaces of the **physical** materials.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*SET_PART
         1
        11
*SET_PART
         2
        22        33
*SET_PART
         3
        44        55        66
*SET_PART
         4
        77
*ALE_MULTI-MATERIAL_GROUP
         1         0   ← 1st line = 1st AMMG ⇒ AMMGID = 1
         2         0   ← 2nd line = 2nd AMMG ⇒ AMMGID = 2
         3         0   ← 3rd line = 3rd AMMG ⇒ AMMGID = 3
         4         0   ← 4th line = 4th AMMG ⇒ AMMGID = 4
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

With this approach, we define only 4 AMMGs (NALEGP = 4). So in LS-PrePost, when plotting the material-group (history variable) contours, we will see 4 colors, one for each material group. One implication is that when the fluids from part 22 and part 33 flow into the same element, they will coalesce and no boundary distinction between them is maintained subsequently. While this may be acceptable for fluids at similar thermodynamic states, this may not be intuitive for solids. For example, if the solid container materials from parts 44, 55 and 66 flow into one element, they will coalesce "like a single fluid", and no interfaces among them are tracked. If this is undesirable, an alternate approach may be taken. It is presented next.

Approach 2: If we want to reconstruct as many interfaces as necessary, in this case, we follow the interface of each part.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_MULTI-MATERIAL_GROUP
         1         1   ←  1st line  =  1st AMMG  ⇒ AMMGID = 1
         2         1   ←  2nd line  =  2nd AMMG  ⇒ AMMGID = 2
         3         1   ←  3rd line  =  3rd AMMG  ⇒ AMMGID = 3
         4         1   ←  4th line  =  4th AMMG  ⇒ AMMGID = 4
         5         1   ←  4th line  =  5th AMMG  ⇒ AMMGID = 5
         6         1   ←  4th line  =  6th AMMG  ⇒ AMMGID = 6
         7         1   ←  4th line  =  7th AMMG  ⇒ AMMGID = 7
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

There are 7 AMMGs in this case (NALEGP = 7). This will involve more computational cost for the additional tracking. Realistically, accuracy will be significantly reduced if there are more than 3 or 4 materials in any one element. In that case, higher mesh resolution may be required.

Example 2:

Oil	Water	Air
Group 1	Group 2	Group 3
Part IDs 1 and 2	Part ID 3	Part IDs 5, 6, and 7

The above example defines a mixture of three groups of materials (or "fluids"), oil, water and air, that is, the number of ALE multi-material groups (AMMGs) NALEGP = 3.

The first group contains two parts (materials), part IDs 1 and 2.

The second group contains one part (material), part ID 3.

The third group contains three parts (materials), part IDs 5, 6 and 7.

*ALE_PRESCRIBED_MOTION

Purpose: Define an imposed nodal motion on a set of multi-material ALE groups for a given time span (if this time frame is short enough, velocity initialization by group can be considered).

Card 1	1	2	3	4	5	6	7	8
Variable	MMSID	INSIDE	SIDR					
Type	I	I	I					
Default	none	0	0					

Remaining cards are optional.†

Translational Velocity Card. Optional card for the translational velocity (see Remark 1).

Card 2	1	2	3	4	5	6	7	8
Variable	LCVTX	LCVTY	LCVTZ					
Type	I	I	I					
Default	Rem 1	Rem 1	Rem 1					

Rotational Velocity Card. Optional card for the rotational velocity (see Remark 1).

Card 3	1	2	3	4	5	6	7	8
Variable	LCVRX	LCVRY	LCVRZ					
Type	I	I	I					
Default	Rem 1	Rem 1	Rem 1					

Center of Rotation Card. Optional card for defining a center of rotation.

Card 4	1	2	3	4	5	6	7	8
Variable	XG	YG	ZG					
Type	F	F	F					
Default	Rem 4	Rem 4	Rem 4					

VARIABLE	DESCRIPTION
MMSID	Multi-Material Set ID (see *SET_MULTI-MATERIAL_GROUP_-LIST).
INSIDE	Flag to define which nodes the motion is prescribed for (see Remark 2): EQ.0: Nodes connected to at least one ALE element that is at the minimum partially filled by a group of MMSID EQ.1: Nodes connected to at least one ALE element that is fully filled by a group of MMSID EQ.2: Nodes only connected to ALE elements that are fully filled by a group of MMSID.
SIDR	Flag controlling the use of this keyword during dynamic relaxation: EQ.0: the keyword is applied during the normal analysis phase only. EQ.1: the keyword is applied during the dynamic relaxation phase but not the normal analysis phase. EQ.2: the keyword is applied during both the dynamic relaxation and the normal analysis phases.
LCVTX LCVTY LCVTZ	Curve IDs for the translation in each global direction; see *DEFINE_CURVE. See Remark 3.
LCVRX LCVRY LCVRZ	Curve IDs for the rotation around each global direction; see *DEFINE_CURVE. See Remark 3.
XG, YG, ZG	Position of the rotation center. See Remark 4.

Remarks:

1. **Zero Velocity.** If only the 1st card is defined, the velocities of the nodes inside ALE groups defined by MMSID are set to zero.

2. **INSIDE.** Depending on the flag INSIDE, the motion is applied to nodes connected to elements partially or fully filled by ALE groups of the set MMSID. For INSIDE = 0, when an element is partially filled by one of these groups, the motion of all its nodes will be prescribed. INSIDE = 1 is more restrictive and only selects nodes connected to at least one element fully filled by a group in MMSID. INSIDE = 2 is even more selective: nodes connected to only fully filled elements are considered. So INSIDE = 1 can apply velocities to nodes at the boundary between fully filled and "empty" elements while INSIDE = 2 only selects nodes inside the ALE group surrounded by fully filled elements.

3. **Birth and Death Times.** The smallest and largest times in the curves are the birth and death times of the related motions. If these time frames are equal to (or slightly larger than) the time step, the related velocities can be initialized during the first cycle.

4. **Rotation Center.** If the rotation center (XG, YG, ZG) is undefined (no 4th card), the rotation is applied around an axis (defined by LCVRX, LCVRY, LCVRZ) passing through the center of mass of the groups defined by MMSID. In an ALE 2D model, this center is shifted to the Y-axis. If the card is included, but has blank fields, then the blank field values default 0.0.

*ALE_REFERENCE_SYSTEM_CURVE

Purpose: This command defines a motion and/or a deformation prescribed for a geometric entity, where a geometric entity may be any part, part set, node set, or segment set. The motion or deformation is completely defined by the 12 parameters shown in the equation below. These 12 parameters are defined in terms of 12 load curves. This command is required only when PRTYPE = 3 in the *ALE_REFERENCE_SYSTEM_GROUP command.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	LCID1	LCID2	LCID3	LCID4	LCID5	LCID6	LCID7	LCID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

Card 3	1	2	3	4	5	6	7	8
Variable	LCID9	LCID10	LCID11	LCID12				
Type	I	I	I	I				
Default	none	none	none	none				

VARIABLE	DESCRIPTION
ID	Curve group ID.
LCID1, …, LCID12	Load curve IDs.

Remarks:

1. The velocity of a node at coordinate (x, y, z) is defined as:

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_5 \\ f_9 \end{Bmatrix} + \begin{bmatrix} f_2 & f_3 & f_4 \\ f_6 & f_7 & f_8 \\ f_{10} & f_{11} & f_{12} \end{bmatrix} \begin{Bmatrix} x - \mathrm{XC} \\ y - \mathrm{YC} \\ z - \mathrm{ZC} \end{Bmatrix}$$

where $f_1(t)$ is the value of load curve LCID1 at time t, $f_2(t)$ is the value of load curve LCID2 at time t and so on. The functions $f_1(t), f_5(t)$, and $f_9(t)$ respectively correspond to the translation components in global x, y, and z direction, while the functions $f_2(t), f_7(t)$, and $f_{12}(t)$ correspond to and expansion or contraction along the x, y, and z axes.

The parameters XC, YC and ZC from the second data card of *ALE_REFERENCE_SYSTEM_GROUP define the center of rotation and expansion of the mesh. If the mesh translates, the center position is updated with $f_1(t), f_5(t)$, and $f_9(t)$.

If LCID8, LCID10, LCID3 are equal to -1, their corresponding values $f_8(t), f_{10}(t)$, and $f_3(t)$ will be equal to $-f_{11}(t)$, $-f_4(t)$, and $-f_6(t)$ so as to make a skew symmetric matrix thereby inducing a rigid rotation of the mesh about the axis **v** defined by the triple,

$$\mathbf{v} = \left(f_{11}(t), f_4(t), f_6(t)\right)$$

Example:

Consider a motion that consists of translation in the x and y direction only. Thus only $f_1(t)$ and $f_5(t)$ are required. Hence only 2 load curve ID's need be defined:

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*ALE_REFERENCE_SYSTEM_GROUP
$       SID     STYPE      PRTYP       PRID     BCTRAN      BCEXP      BCROT     ICOORD
          1         0          3         11          0          7          0
$        XC        YC         ZC    EXPLIM
          0         0          0          0
*ALE_REFERENCE_SYSTEM_CURVE
$ CURVESID
         11
$     LCID1      LCID2      LCID3      LCID4      LCID5      LCID6      LCID7     LCID8
        111          0          0          0        222          0          0         0
$     LCID9     LCID10     LCID11     LCID12
          0          0          0          0
*DEFINE_CURVE
$      lcid       sidr        sfa        sfo       offa       offo     dattyp
        111
$                   a1                    o1
                  0.00                   5.0
                  0.15                   4.0
*DEFINE_CURVE
$      lcid       sidr        sfa        sfo       offa       offo     dattyp
        222
$                   a1                    o1
                  0.00                  -1.0
```

```
                0.15                -5.0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_REFERENCE_SYSTEM_GROUP

Purpose: Associate a geometric entity to a reference system type. A geometric entity may be any part, part set, node set, or segment set of a model (or a collection of meshes). A reference system type refers to the possible transformation allowed for a geometric entity (or mesh). This command defines the type of reference system or transformation that a geometric entity undergoes. In other words, it prescribes how a specified mesh can translate, rotate, expand, contract, be fixed in space, etc.

Card 1	1	2	3	4	5	6	7	8
Variable	SID	STYPE	PRTYPE	PRID	BCTRAN	BCEXP	BCROT	ICR/NID
Type	I	I	I	I	I	I	I	I
Default	none	0	0	0	0	0	0	0

Card 2	1	2	3	4	5	6	7	8
Variable	XC	YC	ZC	EXPLIM	EFAC		FRCPAD	IEXPND
Type	F	F	F	F	F		F	I
Default	0.0	0.0	0.0	∞	0.0		0.1	0

Remaining cards are optional.†

Card 3	1	2	3	4	5	6	7	8
Variable	IPIDXCL	IPIDTYP						
Type	I	I						
Default	0	0						

VARIABLE	DESCRIPTION
SID	Set ID
STYPE	Set type: EQ.0: part set EQ.1: part EQ.2: node set EQ.3: segment set
PRTYPE	Reference system type (see Remark 1): EQ.0: Eulerian EQ.1: Lagrangian EQ.2: Normal ALE mesh smoothing EQ.3: Prescribed motion following load curves; see *ALE_REFERENCE_SYSTEM_CURVE. EQ.4: Automatic mesh motion following mass weighted average velocity in ALE mesh EQ.5: Automatic mesh motion following a local coordinate system defined by three user defined nodes; see *ALE_REFERENCE_SYSTEM_NODE. EQ.6: Switching in time between different reference system types; see *ALE_REFERENCE_SYSTEM_SWITCH. EQ.7: Automatic mesh expansion in order to enclose up to twelve user defined nodes; see *ALE_REFERENCE_SYSTEM_NODE. EQ.8: Mesh smoothing option for shock waves, where the element grid contracts in the vicinity of the shock front: this may be referred to as the Delayed-ALE option. It controls how much the mesh is to be moved during the remap step. This option requires the definition of the 5th parameter Card 2, EFAC; see below for definition. EQ.9: Allowing the ALE mesh(es) to: 1. Translate and/or rotate to follow a local Lagrangian reference coordinate system (whose *ALE_REFERENCE_SYSTEM_NODE card ID is defined by the BCTRAN parameter)

VARIABLE	DESCRIPTION
	2. Expand or contract to enclose a Lagrangian part set ID defined by the PRID parameter. 3. Has a Lagrangian node ID be defined by the ICR/NID parameter to be the center of the ALE mesh expansion.
PRID	A parameter giving additional information depending on the reference system (PRTYPE) choice: PRTYPE.EQ.3: PRID defines a load curve group ID specifying an *ALE_REFERENCE_SYSTEM_CURVE card for mesh translation. This defines up to 12 curves which prescribe the motion of the system. PRTYPE.EQ.4: PRID defines a node set ID (*SET_NODE), for which a mass average velocity is computed. This velocity controls the mesh motion. PRTYPE.EQ.5: PRID defines a node group ID specifying an *ALE_REFERENCE_SYSTEM_NODE card, via which, three nodes forming a local coordinate system are defined. PRTYPE.EQ.6: PRID defines a switch list ID specifying an *ALE_REFERENCE_SYSTEM_SWITCH card. This defines the switch times and the reference system choices for each time interval between the switches. PRTYPE.EQ.7: PRID defines a node group ID specifying an *ALE_REFERENCE_SYSTEM_NODE card. Up to 12 nodes in space forming a region to be enveloped by the ALE mesh are defined. PRTYPE.EQ.9: PRID defines a Lagrangian part set ID (PSID) defining the Lagrangian part(s) whose range of motion is to be enveloped by the ALE mesh(es). This is useful for airbag modeling.
If PRTYPE = 4 or PRTYPE = 5, then	
BCTRAN	BCTRAN is a translational constraint (Remark 3). EQ.0: no constraints EQ.1: constrained x translation

VARIABLE	DESCRIPTION
	EQ.2: constrained y translation
	EQ.3: constrained z translation
	EQ.4: constrained x and y translation
	EQ.5: constrained y and z translation
	EQ.6: constrained z and x translation
	EQ.7: constrained x, y, and z translation
Else If PRTYPE = 9, then	
BCTRAN	BCTRAN is a node group ID from a *ALE_REFERENCE_SYSTEM_NODE card prescribing a local coordinate system (3 node IDs) whose motion is to be followed by the ALE mesh(es).
Else	
BCTRAN	Ignored
End if	
BCEXP	For PRTYPE = 4 and 7, BCEXP is an expansion constraint; otherwise it is ignored (Remark 3).
	EQ.0: no constraints
	EQ.1: constrained x expansion
	EQ.2: constrained y expansion
	EQ.3: constrained z expansion
	EQ.4: constrained x and y expansion
	EQ.5: constrained y and z expansion
	EQ.6: constrained z and x expansion
	EQ.7: constrained x, y, and z expansion
BCROT	BCROT is a rotational constraint (Remark 3). It is ignored unless PRTYPE = 4.
	EQ.0: no constraints
	EQ.1: constrained x rotation
	EQ.2: constrained y rotation

VARIABLE	DESCRIPTION
	EQ.3: constrained z rotation EQ.4: constrained x and y rotation EQ.5: constrained y and z rotation EQ.6: constrained z and x rotation EQ.7: constrained x, y, and z rotation
If PRTYPE = 4	
ICR/(NID)	ICR is a flag the specifies the method LS-DYNA uses for determining the center point for expansion and rotation (Remark 3). EQ.0: The center is at center of gravity of the ALE mesh. EQ.1: The center is at (XC,YC,ZC), just a point in space (it does not have to be a defined node)
Else if PRTYPE = 9	
(ICR)/NID	NID sets the Lagrangian node ID for the node that anchors the center of ALE mesh expansion (Remark 2).
End if	
XC, YC, ZC	Center of mesh expansion and rotation for PRTYPE = 4; otherwise ignored. See ICR above.
EXPLIM	Limit ratio for mesh expansion and contraction. Each cartesian direction is treated separately. The distance between the nodes is not allowed to increase by more than a factor EXPLIM or decrease to less than a factor 1/EXPLIM. This flag applies only for PRTYPE = 4, otherwise it is ignored.
EFAC	Mesh remapping factor for PRTYPE = 8 only, otherwise it is ignored. EFAC ranges between 0.0 and 1.0. When EFAC approaches 1.0, the remapping approaches the Eulerian behavior. The smaller the value of EFAC, the closer the mesh will follow the material flow in the vicinity of a shock front, that is, approaching Lagrangian behavior. Note that excessively small values for EFAC can result in severe mesh distortions as the mesh follows the material flow. As time evolves, the mesh smoothing behavior will approach that of an

VARIABLE	DESCRIPTION
	Eulerian system.
FRCPAD	For PRTYPE = 9 this is an ALE mesh padding fraction, otherwise it is ignored. FRCPAD ranges from 0.01 to 0.2. If the characteristic Lagrange mesh dimension, dL_1, exceeds $$(1 - 2 \times \text{FRCPAD}) \times dL_A,$$ where dL_A is the characteristic length of the ALE mesh, then the ALE mesh is expanded so that $$dL_A = \frac{dL_1}{1 - 2 \times \text{FRCPAD}}.$$ This provides an extra few layers of ALE elements beyond the maximum Lagrangian range of motion. EQ.0.01: $dL_A = {dL_L}/{0.98} = dL_L \times 1.020408$ EQ.0.20: $dL_A = {dL_L}/{0.60} = dL_L \times 1.666667$
IEXPND	For PRTYPE = 9, this is an ALE mesh expansion control flag, otherwise it is ignored. EQ.0: Both mesh expansion and contraction are allowed. EQ.1: Only mesh expansion is allowed.
IPIDXCL	An ALE set ID to be excluded from the expansion and/or contraction only. Translation and rotation are allowed. For example, this may be used to prevent the ALE mesh (or part) at the inflator gas inlet region from expanding too much. High ALE mesh resolution is usually required to resolve the high speed flow of the gas into the airbag via point sources. See Remark 2.
IPIDTYPE	Set ID type of IPIDXCL: EQ.0: PSID EQ.1: PID

Remarks:

1. **Required associated cards.** Some PRTYP values may require a supplemental definition defined using the corresponding PRID. For example, PRTYP = 3 requires a *ALE_REFERENCE_SYSTEM_CURVE card. If PRID = n, then in the

corresponding *ALE_REFERENCE_SYSTEM_CURVE card, ID = *n*. Similar association applies for any PRTYP (i.e. 3, 5, 6, or 7) which requires a definition for its corresponding PRID parameter.

2. **Mesh centering.** For PRTYPE = 9, ICR/NID can be used to keep a high-density ALE mesh centered on the region of greatest interest, such as the inflator orifices region in an airbag model. For example, in the case of nonsymmetrical airbag deployment, assuming that the ALE mesh is initially finer near the inlet orifices and gradually coarsened away from it, defining an "anchor node" at the center of the orifice location will keep the fine ALE mesh region centered on the orifice region. Then this fine ALE mesh region will not be shifted away (from the point sources) during expansion and translation. The ALE mesh can move and expand outward to envelop the Lagrangian airbag in such a way that the inlet is well resolved throughout the deployment.

3. **Additional constraints.** The table below shows the applicability of the various choices of PRTYPE. Simple deductions from the functional definitions of the PRTYPE choices will clarify the applications of the various constraints. For example, when PRTYP = 3, nodal motion of the ALE mesh is completely controlled by the 12 curves. Therefore, no constraints are needed.

PRTYPE	ICR/NID	BCTRAN	BCROT	BCEXP
3	NO	NO	NO	NO
4	YES (ICR)	YES	YES	YES
5	NO	YES	NO	NO
6	NO	NO	NO	NO
7	NO	NO	NO	YES
8	NO	NO	NO	NO
9	YES (NID)	YES (NGID)	NO	NO

Example 1:

Consider a bird-strike model containing 2 ALE parts: a bird surrounded by air (or void). Part set ID 1 is defined containing both parts. To allow for the meshes of these 2 parts to move with their combined mass-weighted-average velocity, PRTYPE = 4 is used. Note that BCEXP = 7 indicating mesh expansion is constrained in all global directions.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*ALE_REFERENCE_SYSTEM_GROUP
```

```
$      SID     STYPE      PRTYP       PRID    BCTRAN     BCEXP     BCROT    ICOORD
         1         0          4          0         0         7         0
$       XC        YC         ZC    EXPLIM
         0         0          0          0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

Example 2:

Consider a bouncing ball model containing 2 ALE parts: a solid ball (PID 1) surrounded by air or void (PID 2). Part set ID 1 is defined containing both parts. To allow for the meshes of these 2 parts to move with 2 reference system types: (a) first, they move with their combined mass-weighted-average velocity between 0.0 and 0.01 second; and subsequently (between 0.01 and 10.0 seconds) their reference system is switched to (b) an Eulerian system (thus the mesh is fixed in space), a reference system "SWITCH" is required. This is done by setting PRTYPE = 6. This PRTYPE requires a corresponding *ALE_REFERENCE_SYSTEM_SWITCH card. Note that PRID = 11 in the *ALE_REFERENCE_SYSTEM_GROUP card corresponds to the SWITCHID = 11 in *ALE_REFERENCE_SYSTEM_SWITCH card.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_REFERENCE_SYSTEM_GROUP
$      SID     STYPE      PRTYP       PRID    BCTRAN     BCEXP     BCROT    ICOORD
         1         0          6         11         0         7         7
$       XC        YC         ZC    EXPLIM   EULFACT SMOOTHVMX
         0         0          0          0       0.0
*ALE_REFERENCE_SYSTEM_SWITCH
$ SWITCHID
        11
$       t1        t2         t3         t4         t5         t6         t7
      0.01      10.0
$    TYPE1     TYPE2      TYPE3      TYPE4      TYPE5      TYPE6      TYPE7      TYPE8
         4         0
$      ID1       ID2        ID3        ID4        ID5        ID6        ID7        ID8
         0         0          0          0          0          0          0          0
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

*ALE_REFERENCE_SYSTEM_NODE

Purpose: Define a group of nodes that control the motion of an ALE mesh. This keyword is used *only* when PRTYPE = 5 or 7 in a corresponding *ALE_REFERENCE_SYSTEM_-GROUP card.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	NID1	NID2	NID3	NID4	NID5	NID6	NID7	NID8
Type	I	I	I	I	I	I	I	I

Enclosing Nodes Card. This card is only included if PRTYPE = 7.

Card 3	1	2	3	4	5	6	7	8
Variable	NID9	NID10	NID11	NID12				
Type	I	I	I	I				

VARIABLE	DESCRIPTION
ID	Node group ID for PRTYPE 5 or 7; see PRID on *ALE_REFERENCE_SYSTEM_GROUP.
NID1, …, NID12	User specified nodes. See Remarks 1 and 2.

Remarks:

1. **ALE Motion from Local Coordinate System.** For PRTYPE = 5, the ALE mesh is forced to follow the motion of a coordinate system, which is defined by three

nodes (NID1,NID2,NID3). These nodes are located at x_1, x_2, and x_3, respectively. The axes of the coordinate system, x', y', and z', are defined as:

$$x' = \frac{x_2 - x_1}{|x_2 - x_1|}$$
$$y' = z' \times x'$$
$$z' = x' \times \frac{x_3 - x_1}{|x' \times (x_3 - x_1)|}$$

Note that $x_1 \rightarrow x_2$ is the local x'axis, $x_1 \rightarrow x_3$ is the local y' axis and x' crosses y' gives the local z' axis. These 3 nodes are used to locate the reference system at any time. Therefore, their positions relative to each other should be as close to an orthogonal system as possible for better transformation accuracy of the ALE mesh.

2. **ALE Motion from Enclosing Nodes.** For PRTYPE = 7, the ALE mesh is forced to move and expand, so as to enclose up to twelve user defined nodes (NID1, …, NID12). This is a rarely used option.

Example 1:

Consider modeling the sloshing of water inside a rigid tank. Assuming there are 2 ALE parts, the water (PID 1) and air or void (PID 2) contained inside a rigid (Lagrangian) tank (PID 3). The outer boundary nodes of both ALE parts are merged with the inner tank nodes. Part set ID 1 is defined containing both ALE parts (PIDs 1 and 2). To allow for the meshes of the 2 ALE parts to move with the rigid Lagrangian tank, PRTYPE = 5 is used. The motion of the ALE parts then follows 3 reference nodes on the rigid tank. These 3 reference nodes must be defined by a corresponding *ALE_REFERENCE_SYSTEM_NODE card. In this case the reference nodes have the nodal IDs of 5, 6 and 7. Note that PRID = 12 in the *ALE_REFERENCE_SYSTEM_GROUP card corresponds to the SID = 12 in the *ALE_REFERENCE_SYSTEM_NODE card.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*ALE_REFERENCE_SYSTEM_GROUP
$       SID     STYPE      PRTYP       PRID    BCTRAN     BCEXP     BCROT    ICOORD
          1         0          5         12
$        XC        YC         ZC    EXPLIM
          0         0          0          0
*ALE_REFERENCE_SYSTEM_NODE
$     NSID
         12
$        N1        N2         N3         N4         N5         N6         N7         N8
          5         6          7
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

*ALE_REFERENCE_SYSTEM_SWITCH

Purpose: Time-dependent switching between the reference system types for ALE geometric entities. Multiple switches can occur during the simulation. See PRTYPE in *ALE_REFERENCE_SYSTEM_GROUP. This command is required only when PRTYPE = 6 in *ALE_REFERENCE_SYSTEM_GROUP. See Example 2 in the *ALE_REFERENCE_SYSTEM_GROUP.

Card 1	1	2	3	4	5	6	7	8
Variable	ID							
Type	I							
Default	none							

Card 2	1	2	3	4	5	6	7	8
Variable	T1	T2	T3	T4	T5	T6	T7	
Type	F	F	F	F	F	F	F	
Default	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Card 3	1	2	3	4	5	6	7	8
Variable	TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6	TYPE7	TYPE8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Card 4	1	2	3	4	5	6	7	8
Variable	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

VARIABLE	DESCRIPTION
ID	Switch list ID; see *ALE_REFERENCE_SYSTEM_GROUP.
T1, …, T7	Times for switching reference system type. By default, the reference system TYPE1 occurs between time = 0.0 and time = T1, and TYPE2 occurs between time = T1 and time = T2, etc.
TYPE1, …, TYPE8	Reference system types (also see PRTYPE under *ALE_REFERENCE_SYSTEM_GROUP): EQ.0: Eulerian EQ.1: Lagrangian EQ.2: Normal ALE mesh smoothing EQ.3: Prescribed motion following load curves; see *ALE_REFERENCE_SYSTEM_CURVE. EQ.4: Automatic mesh motion following mass weighted average velocity in ALE mesh EQ.5: Automatic mesh motion following a local coordinate system defined by three user defined nodes; see *ALE_REFERENCE_SYSEM_NODE.
ID1, …, ID8	The corresponding PRID parameters supporting each PRTYPE used during the simulation.

Remarks:

1. **Switching Time.** The beginning time is assumed to be t = 0.0, and the starting PRTYPE is TYPE1. So at T1, the 1st switching time, PRTYPE is switched from TYPE1 to TYPE2, and so forth. This option can be complex in nature, so it is seldom applied.

*ALE_REFINE

See *CONTROL_REFINE_ALE.

*ALE_SMOOTHING

Purpose: This smoothing constraint either keeps a dependent ALE node at its initial parametric location along a line between two other ALE nodes (see Figure 4-14) or causes a dependent ALE node to follow the motion of two non-ALE nodes (see Remark 1). This constraint is active during each mesh smoothing operation. This keyword can be used with ALE solids, ALE shells, and ALE beams.

Card 1	1	2	3	4	5	6	7	8
Variable	DNID	NID1	NID2	IPRE	XCO	YCO	ZCO	
Type	I	I	I	I	F	F	F	
Default	none	none	none	0	0.0	0.0	0.0	

VARIABLE	DESCRIPTION
DNID	Dependent node or node set ID: GT.0: DNID is an ALE node. EQ.0: The dependent nodes are the nodes of an ALE mesh connected to the nodes in INID1. See Remark 2. LT.0: -DNID is the ID of an ALE node set. See Remark 2.
NID1	ID of first node or set for constraining the dependent nodes: GT.0: NID1 is a node. LT.0: -NID1 is a segment set ID if XCO = YCO = ZCO = 0.0. Otherwise, -NID1 is a node set ID. See Remark 2.
NID2	ID of second node or node set for constraining the dependent nodes: GT.0: NID2 is a node. EQ.0: The dependent node motion is solely controlled by NID1. See Remarks 2 and 3. LT.0: -NID2 is a node set ID. See Remark 2.
IPRE	EQ.0: Smoothing constraints are performed after mesh relaxation.

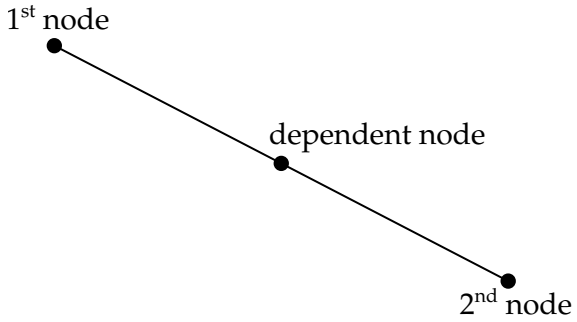

Figure 4-14. This simple constraint, which ensures that a constrained node remains on a straight line between two controlling nodes, is sometimes necessary during ALE smoothing.

VARIABLE	DESCRIPTION
	EQ.1: Smoothing constraints are performed before mesh relaxation.
XCO	x-coordinate of constraint vector
YCO	y-coordinate of constraint vector
ZCO	z-coordinate of constraint vector

Remarks:

1. **When NID1 and NID2 Nodes are not ALE Nodes.** If DNID, NID1 and NID2 are ALE nodes, the positions of NID1 and NID2 are interpolated to position DNID. If NID1 and NID2 are *not* ALE nodes, the motions of NID1 and NID2 are interpolated to move DNID.

2. **Constraint Generation.** If NID1 is a set ID, DNID and NID2 should be node set IDs or zeros. In this case, the constraints are created during initialization and printed out to a file called alesmoothingenerated.k.

   If NID1 is a node set, the constraints associated with a given node in NID1 are generated by finding the closest dependent nodes to an axis passing through the NID1 node and oriented by the constraint vector (XCO,YCO,ZCO). If NID1 is a segment set, each node in the given segments can constrain dependent nodes. The axis direction for a node is found by averaging all the normals of the segments connected to the node.

   If DNID = 0, NID1 should be a set of nodes or segments along the boundary of an ALE mesh. For a given node in NID1 or in a segment of NID1, constraints are created for all the nodes of the mesh found the closest to the axis described previously. The search for dependent nodes starts with nodes of elements

connected to the node in NID1 and stops when a boundary node with an element connectivity similar to that of the node in NID1 is reached or when a node in the set NID2 (if defined) is found.

3. **NID2 = 0 and DNID ≠ 0.** If NID2 = 0 and DNID is nonzero, NID1 should *not* be an ALE node or set. Otherwise, the positions of DNID and INID1 would match and the element volumes between them could be zero or negative. Only DNID and INID1 motion should match in such a case.

***ALE_STRUCTURED_FSI**_{OPTION1}_{OPTION2}

Purpose: Perform Fluid-Structure Interaction (FSI) between Lagrangian structures modeled by shell/solids and ALE multi-material fluids using a structured ALE mesh (*ALE_STRUCTURED_MESH).

Available options for *OPTION1* include:

<BLANK>

ABEXT

ABINT

ABEXT invokes special logic for the exterior parts of airbags, while ABINT invokes special logic for interior parts.

Available options for *OPTION2* include:

<BLANK>

TITLE

With the keyword option TITLE, you can set a coupling (card) ID number and title for each coupling card. If a title is not defined, LS-DYNA will automatically create an internal title for this coupling definition.

Comparison to *CONSTRAINED_LAGRANGE_IN_SOLID:

*ALE_STRUCTURED_FSI, as its name suggests, only works with the structured ALE solver. It mostly follows the format of *CONSTRAINED_LAGRANGE_IN_SOLID, but with a few exceptions:

- **Coupling Type.** Unlike *CONSTRAINED_LAGRANGE_IN_SOLID, *ALE_STRUCTURED_FSI only has one penalty formulation coupling method which is similar to a combination of CTYPE = 4 and 5 in *CONSTRAINED_LAGRANGE_IN_SOLID.
- **Number of Coupling Points.** For each Lagrangian segment, there are a certain number of coupling points, evenly distributed at the segment surface. Penalty springs are attached to those coupling points. Using *CONSTRAINED_LAGRANGE_IN_SOLID requires specifying the number while with this keyword, LS-DYNA automatically determines the number of coupling points during the initialization phase.
- **Leakage Control.** Leakage control is automated when using *ALE_STRUCTURED_FSI. Fluid leakage is detected and cured automatically with no user intervention needed.

- **Normal Type.** Normal type selection is automated, based on the local geometry. Users do not need to choose between nodal/segment normal.
- **Edge Coupling.** Edge coupling is automatic. Shell segments are picked out and the exposed edges are coupled. *CONSTRAINED_LAGRANGE_IN_SOLID_-EDGE is not needed.
- **Erosion Coupling.** Eroded solid elements will change the coupling segments. Segments belonging to eroded elements need to be deleted, and newly exposed segments need to be added. With *CONSTRAINED_LAGRANGE_IN_SOLID, CTYPE = 5 needs to be specified in order to activate this segment's modification algorithm. In the new *ALE_STRUCTURED_FSI, this process is always on and no flag is needed.

Title Card. Additional card for the TITLE keyword option.

Card ID	1	2	3	4	5	6	7	8
Variable	COUPID	TITLE						
Type	I	A70						

Card 1	1	2	3	4	5	6	7	8
Variable	LSTRSID	ALESID	LSTRSTYP	ALESTYP				MCOUP
Type	I	I	I	I				I
Default	none	none	0	0				0

Card 2	1	2	3	4	5	6	7	8
Variable	START	END	PFAC	FRIC	ILVL	FLIP	DECAY	OFFSET
Type	F	F	F	F	I	I	F	F
Default	0.0	10^{10}	0.1	0.0	0	0	0.0	0.0

VARIABLE	DESCRIPTION
COUPID	Coupling (card) ID. If not defined, LS-DYNA will assign an internal coupling ID based on the order of appearance in the input deck.
TITLE	Description of this coupling definition (A70)
LSTRSID	Set ID defining a part, part set, or segment set ID of the Lagrangian structure (see *PART, *SET_PART, or *SET_SEGMENT)
ALESID	Set ID defining a part or part set ID of the structured ALE mesh (see *PART)
LSTRSTYP	Set type of LSTRSID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID) EQ.2: Segment set ID (SGSID)
ALESTYP	Set type of ALESID: EQ.0: Part set ID (PSID) EQ.1: Part ID (PID)
MCOUP	Multi-material(s) to be coupled (see Remark 1): EQ.0: Couple with all multi-material groups. EQ.-N: -N is the ID of *SET_MULTI-MATERIAL_GROUP_LIST.
START	Start time for coupling
END	End time for coupling
PFAC	Penalty factor. PFAC is a scale factor for scaling the estimated stiffness of the interacting (coupling) system. It is used to compute the coupling forces to be distributed on the Lagrangian and ALE parts. GT.0: Fraction of estimated critical stiffness LT.0: PFAC must be an integer, and -PFAC is a load curve ID. The curve defines the coupling pressure on the y-axis as a function of the penetration along the x-axis.
FRIC	Friction coefficient. The friction force is evaluated as the normal force multiplied by the friction coefficient. GT.0: Constant friction coefficient

VARIABLE	DESCRIPTION
	EQ.-N: Variable friction coefficient defined by a table with ID N. The friction coefficient is a function of coupling pressure and relative velocity. The table uses the coupling pressure as the parameter values. See Examples.
ILVL	Number of airbag segments to look downstream from the gas front. It is only used for the ABEXT keyword option. Together with DECAY, it activates pre-opening logic for gas-front segments to alleviate constrained flow typically observed in gas flow through long tubes. Please refer to *AIRBAG_SALE (specifically Remark 8) for a detailed explanation.
FLIP	A Lagrangian segment will couple to fluid on only one side of the segment. The segment normal is assumed to point toward the fluids to which it is coupled. If that is not the case, set FLIP to 1. EQ.0: No action EQ.1: Flip the segment normal, so it points toward the fluids to which it is to be coupled.
DECAY	Decaying factor. It is only used for the ABEXT keyword option. Together with ILVL, it activates pre-opening logic for gas-front segments to alleviate constrained flow typically observed in gas flow through long tubes. Please refer to *AIRBAG_SALE (specifically Remark 8) for a detailed explanation.
OFFSET	Specofy how much to offset Lagrange segments away from the segment surface: EQ.0.0: No offset GT.0.0: Fraction of Lagrange segment thickness LT.0.0: OFFSET must be an integer, and -OFFSET is a load curve ID. The curve defines the offset distance in absolute value on the y-axis as a function of the time along the x-axis.

Remarks:

1. **Multi-material coupling option.** MCOUP is used to specify which ALE multi-materials are to be coupled. In a typical simulation, we want to prevent some fluid(s) on one side of the structure from penetrating through to the other side. In this case, we would pick the AMMGs on one side and list them under *SET_MULTI-MATERIAL_GROUP_LIST.

Coupling to all materials as activated by MCOUP = 0 is generally not recommended. LS-DYNA calculates the fluid coupling interface as the surface where the sum of coupled ALE materials occupies a volume fraction equal to 50%. Since MCOUP = 0 couples to all materials, the sum of all coupled ALE materials is, in this case, trivially 100%. Consequently, when MCOUP = 0, there will not be a fluid interface with which to track leakage.

2. **Output.** To obtain the sliding energy output for each FSI, specify a *DATABASE_SLEOUT in the input deck. Including this keyword causes a file called slefsi to be output. The format of slefsi follows sleout exactly.

Examples:

The following is a partial input deck that illustrates defining a variable friction table. Table 2 gives three coupling pressures that correspond to the three *DEFINE_CURVEs specified under the table definition (see *DEFINE_TABLE for details). Each curve specifies the relationship between relative velocity and friction coefficient for a given coupling pressure. For example, if the coupling pressure is 100, curve 4 is used. If coupling pressure = 50, both curves 3 and 4 are used. The friction coefficient is interpolated from the values given by the curves.

```
*ALE_STRUCTURED_FSI
$# lstrsid    alesid   lstrtyp   alestyp         -         -         -     mcoup
         1         1         0         1         0         0         0         -
1
$#   start       end      pfac      fric         -      flip
                         0.1        -2         0         0
*DEFINE_TABLE
$     tbid       sfa      offa
         2
$                  p
                 0.0
               100.0
              1000.0
$---------------------------------------------------------------------------
--
*DEFINE_CURVE
$ curve for p = 0.0
$     lcid      sidr       sfa       sfo      offa      offo    dattyp     lcint
         3         0       1.0       1.0       0.0       0.0         0         0
$               vel1                  mu1
                 0.0                  0.6
    1.0000000000e+02                  0.5
*DEFINE_CURVE
$ curve for p = 100.0
$     lcid      sidr       sfa       sfo      offa      offo    dattyp     lcint
         4         0       1.0       1.0       0.0       0.0         0         0
$               vel2                  mu2
                 0.0                  0.5
    1.0000000000e+02                  0.4
*DEFINE_CURVE
$ curve for p = 1000.0
```

```
$     lcid      sidr       sfa       sfo      offa      offo    dattyp     lcint
         5         0       1.0       1.0       0.0       0.0         0         0
$             vel3                 mu3
              0.0                  0.3
    1.0000000000e+02               0.2
```

*ALE_STRUCTURED_MESH

Purpose: Generate a structured 2D or 3D mesh and invoke the Structured ALE (S-ALE) solver. Spacing parameters are input through one or more of the *ALE_STRUCTURED_MESH_CONTROL_POINTS cards. The local coordinate system is defined using the *ALE_STRUCTURED_MESH card.

In certain contexts, it is advantageous to use structured meshes. With structured meshes, the element and node connectivity are straightforward, and the searching algorithm used for ALE coupling is greatly simplified. Also, numerous checks are avoided because these meshes include only hex elements.

The S-ALE solver supports SMP, MPP, and MPP hybrid configurations. All three implementations require less simulation time and memory usage than the regular ALE solver. We, therefore, encourage using the S-ALE solver when the ALE mesh is structured.

Once an ALE mesh is generated using an *ALE_STRUCTURED_MESH card, this card invokes the S-ALE solver and performs the ALE advection time step. For fluid-structure interaction, *ALE_STRUCTURED_FSI is recommended over *CONSTRAINED_LAGRANGE_IN_SOLID. *ALE_STRUCTURED_FSI has better, automated leakage detection and control, has a much cleaner and easier input, and is designed for the S-ALE solver.

This keyword can be used multiple times. In each occurrence, an independent, separate mesh is constructed. These meshes can occupy different spatial domains or the same spatial domain. Simulations are executed in those meshes independently.

There are two related keywords: *ALE_STRUCTURED_MESH_TRIM to trim/untrim the generated mesh and *ALE_STRUCTURED_MESH_MOTION to control the mesh motion during the simulation.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID	DPID	NBID	EBID				TDEATH
Type	I	I	I	I				F
Default	none	none	Rem 6	Rem 6				10^{16}

Card 2	1	2	3	4	5	6	7	8
Variable	CPIDX	CPIDY	CPIDZ	NID0	LCSID			
Type	I	I	I	I	I			
Default	none	none	none	none	none			

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID. A unique number must be specified.
DPID	Default Part ID. The elements generated are assigned to DPID. This part definition is automatically generated during the input phase and contains neither material nor element formulation information. Please see Remark 1. This ID also serves as an indicator for mesh merging, that is, if the same part ID is used for several adjacent generated meshes, they will be merged together to form a single mesh consisting of these sub-meshes. Please see Remark 5.
NBID	Nodes are generated and assigned with node IDs starting from NBID. If setting to a nonzero value (recommended), use an ID larger than the maximum existing node ID. If unset or 0 (not recommended), NBID is internally set to the maximum node ID + an offset. See Remark 6.
EBID	Elements are generated and assigned with element IDs starting from EBID. If setting to a nonzero value (recommended), use an ID larger than the maximum existing element ID. If unset or 0 (not recommended), EBID is internally set to the maximum element ID + an offset. See Remark 6.
TDEATH	Death time for this mesh. Please see Remark 3.
CPIDX, CPIDY, CPIDZ	Control point IDs defining node ID/value pairs along each local axis. See *ALE_STRUCTURED_MESH_CONTROL_POINTS. Setting CPIDX to 0 or -1 invokes the ALE to S-ALE converter. Please see Remark 4. Note that for 2D problems CPIDZ is ignored.

VARIABLE	DESCRIPTION
NID0	NID0 sets the mesh's origin node. During the simulation, prescribed motion applied to this node applies to the entire structure S-ALE mesh.
LCSID	Local coordinate system ID. Please see Remark 2.

Remarks:

1. **DPID.** DPID only consists of elements and nodes. It does not include material properties or integration rules. The requirement that a part ID be specified for these automatically generated S-ALE solid elements exists only to satisfy the legacy rule that every element must be associated with a part. You do not need to set up the *PART card for DPID. All part definitions used in this card only refer to the mesh, not the material.

2. **LCSID.** The data cards associated with a *DEFINE_COORDINATE_... keyword define the local coordinate system. This local coordinate system specifies the three cardinal directions used for generating the structured ALE mesh. The structured mesh can be made to rotate by specifying a rotating local coordinate system. To define a rotating local coordinate system, use *DEFINE_COORDINATE_NODES with FLAG = 1 and then apply the prescribed motion to the three coordinate nodes.

3. **TDEATH.** This option provides a way to terminate an ALE calculation while Lagrangian parts still evolve. Once this death time is reached, all elements belonging to this mesh are deleted. In addition to that, ALE FSI cards, including *ALE_STRUCTURED_FSI, *CONSTRAINED_LAGRANGE_IN_SOLID, and *ALE_COUPLING_NODAL_*OPTION*, are checked to see if related ALE elements were deleted. If no ALE elements are left on the fluid side for certain coupling cards, this coupling card is disabled from further calculation as well.

   For example, consider a 2-stage simulation. In the first stage, Lagrange structures are under certain loadings from ALE fluids through ALE/FSI. Later in the second stage, we continue simulating the Lagrange structure behavior while stopping all ALE-related calculations by setting TDEATH to the start time of the 2nd stage.

4. **ALE to S-ALE converter.** For existing ALE models with rectilinear meshes, *ALE_STRUCTURED_MESH can invoke the conversion of the ALE mesh into an S-ALE mesh. To use this feature, add a *ALE_STRUCTURED_MESH card in the model input with CPIDX = -1 or 0 leaving all other fields blank. LS-DYNA will then convert all ALE keywords to the S-ALE format and write the modified input to a file named saleconvrt.inc. The solver used to perform the analysis

depends on the value of CPIDX. If -1, the S-ALE solver is used; if 0, the ALE solver is used. Please note that this converter is only available in 3D.

5. **Merging meshes.** Two or more separated, but adjacent sub-meshes can be merged to form a mesh of complex shape. To do that, first set up two or more sub-meshes; make sure their mesh spacing is the same at their interfaces so that these interface nodes match each other (a tolerance of 1/100 element size is allowed); and finally use the SAME DPID in those *ALE_STRUCTURED_MESH cards. The code automatically checks and merges sub-meshes with the same DPID.

6. **Default behavior of NBID/EBID.** Generally, we recommend not leaving NBID and EBID blank or zero. When left blank or set to 0, the keyword reader scans the input deck to find the maximum user ID and then adds an offset to that maximum ID to generate NBID/EBID. The initial design had the offset set to 1. However, ID collisions began occurring due to other features autogenerating nodes/elements in the same manner. With R16.0, we changed this offset to 1,000,000 to help avoid this issue.

Example:

The following example generates a regular evenly distributed 0.2 by 0.2 by 0.2 box mesh that has 22 nodes along each direction. The generated mesh is aligned to the local coordinate system specified by nodes 2, 3, and 4 with an origin at node 1.

All the elements inside the mesh are assigned to part 1. Note that part 1 is not explicitly defined in the input. The necessary part definition is automatically generated and contains neither material definitions nor integration rules.

```
*ALE_STRUCTURED_MESH
$    mshid      dpid      nbid      ebid                                  tdeath
         1         1    200001    200001                                   0.010
$    cpidx     cpidy     cpidz      nid0     lcsid
      1001      1001      1001         1       234
*DEFINE_COORDINATE_NODES
$      cid      nid1      nid2      nid3      flag
       234         2         3         4         1
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1001
$                 x1                  x2
                   1                  .0
                  22                  .2
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
*END
```

*ALE_STRUCTURED_MESH_CONTROL_POINTS

Purpose: Provide spacing information used by the *ALE_STRUCTURED_MESH keyword to generate a 2D or 3D structured ALE mesh.

Each instance of the *ALE_STRUCTURED_MESH_CONTROL_POINTS card defines a one-dimensional mesh using control points. Each control point consists of a node number (see N in Card 2a/b) and a coordinate (see X in Card 2a/b). The first control point *must* be node 1, and the node number of the last point defines the total number of nodes. Between any two control points, the mesh is uniform or progressive, depending on the input. The *ALE_STRUCTURED_MESH card, in turn, defines a simple three-dimensional mesh from the triple product of three *ALE_STRUCTURED_MESH_CONTROL_POINTS one-dimensional meshes.

Card Summary:

Card 1. This card is required.

CPID		ICASE	SFO		OFFO		

Card 2a. For ICASE = 0, include at least two instantiations of this card to give the control points. A minimum of two cards is required to give the end points and the number of nodes. Input is terminated with the next keyword ("*") card.

N	X	RATIO	

Card 2b. For ICASE = 1 or 2, include at least two instantiations of this card to give the control points. A minimum of two cards is required to give the end points and the number of nodes. Input is terminated with the next keyword ("*") card.

N	X	XL	

Data Card Definitions:

Card 1	1	2	3	4	5	6	7	8
Variable	CPID		ICASE	SFO		OFFO		
Type	I		I	F		F		
Default	none		0	1.		0.		

VARIABLE	DESCRIPTION
CPID	Control Points ID. A unique number must be specified. This ID is to be referred to in the three fields labeled CPIDX, CPIDY, and CPIDZ in *ALE_STRUCTURED_MESH.
ICASE	A flag to trigger special algorithms for a more user-friendly input format for creating progressive mesh spacing: EQ.0: Ratio-based progressive mesh spacing (see Card 2a) EQ.1: Base element size progressive mesh spacing (see Card 2b and Remark 4) EQ.2: Base node progressive mesh spacing (see Card 2b and Remark 5)
SFO	Scale factor for ordinate value. This is useful for simple modifications. EQ.0.0: default set to 1.0.
OFFO	Offset for ordinate values. See Remark 1.

Point Cards for ICASE = 0. Include this card for ICASE = 0. Put one control point per card. Input is terminated at the next keyword ("*") card. *At least two cards are required with one having N = 1.*

Card 2a	1	2	3	4	5	6	7	8
Variable	N		X		RATIO			
Type	I		F		F			
Default	none		none		0.0			

VARIABLE	DESCRIPTION
N	Control point node number
X	Control point position
RATIO	Ratio for progressive mesh spacing. A progressively larger or smaller mesh will be generated between the control point that has a nonzero ratio specified and the control point following it. If a

VARIABLE	DESCRIPTION
	control point and the one following it both have RATIO as zero, the mesh is uniform between them. See Remarks 2 and 3.
	GT.0.0: Mesh size increases
	$dl_{n+1} = dl_n \times (1 + \text{RATIO})$
	LT.0.0: Mesh size decreases
	$dl_{n+1} = dl_n / (1 - \text{RATIO})$

Point Cards for ICASE = 1 or 2. Include this card for ICASE = 1 or 2. Put one control point per card. Input is terminated at the next keyword ("*") card. *At least two cards are required with one having N = 1.* See Remarks 4 and 5 for details about how to use these cards to obtain uniform and progressive meshes. See Remark 3.

Card 2b	1	2	3	4	5	6	7	8
Variable	N		X		XL			
Type	I		F		F			
Default	none		none/optional		0.0/none			

VARIABLE	DESCRIPTION
N	Control point node number
X	Control point position. This is required for ICASE = 1. For ICASE = 2, see Remark 5 for details about when to include this input.
XL	Length of element at the control. For ICASE = 1, see Remark 4 for details about when to enter this value. This input is required for ICASE = 2.

Remarks:

1. **Coordinates scaling.** The ordinate values are scaled after the offsets are applied, that is:

$$\text{Ordinate value} = \text{SFO} \times (\text{Defined value} + \text{OFFO}) .$$

2. **Progressive mesh spacing.** The formula used to calculate element length is as follows:

$$\mathrm{dl}_{\mathrm{base}} = |x_{\mathrm{end}} - x_{\mathrm{start}}| \times (\mathrm{factor} - 1)/(\mathrm{factor}^n - 1)$$

in which $\mathrm{dl}_{\mathrm{base}}$ is the smallest base length; x_{start} and x_{end} are the coordinates of the start and end points, respectively; factor = 1 + RATIO if RATIO > 0 or 1/(1 − RATIO) if RATIO < 0; and n is number of elements. Please note that element size either increases by RATIO if RATIO > 0 or decreases by − RATIO/(1 − RATIO) if RATIO < 0 each time. The overall effect is the same; that is, starting from the smallest element, each time the element size is increased by |RATIO|.

3. **Transition meshes.** If the mesh has different mesh sizes and transition zones between them (where the mesh size progressively increases or decreases), then the ratios and number of elements for these zones are reported in the messag file (look for messages KEY+1174). Updated lists of control points are output in the messag file for each *ALE_STRUCTURED_MESH_CONTROL_POINTS with more than 3 control points.

4. **ICASE = 1.** Assume we want to use progressive mesh spacing for a certain region. We know the number of elements and the overall length. And we have a "base" element size in mind. However, the generic setup requires a "ratio", which needs an iterative solution of a n^{th} order polynomial equation based on the "base" element size. ICASE = 1 aims to simplify the setup. When it is activated, we can input the "base" element size. The value of the "ratio" is internally calculated and used to generate the S-ALE mesh. We will use examples to illustrate this option.

   For this discussion, a region is defined by 2 point cards (2 instantiations of Card 2b). In the example below, *ALE_STRUCTURED_MESH_CONTROL_-POINTS with CPID 1001 has three regions. The first region spans [0, 0.0755]; the second one [0.0755, 0.1245]; and the third one [0.1245, 0.2].

   The RATIO field in the generic setup is now marked as XL. In this case, it specifies the element size at that location. The rule is simple: a region can have a total of 0, 1, or 2 nonzero XL values specified at its two ends. 1 occurrence causes progressive mesh spacing. 0 or 2 occurrences cause an evenly spaced mesh. Please note that when 2 values of XL are given, they must have the same value since the mesh is evenly spaced. Otherwise, the input is not logically correct.

   In the example below:

   a) Region 1 has progressive spacing, and the preferred element size at the 8^{th} node is 0.0070.

   b) In region 2, the mesh is evenly distributed.

c) Region 3 has progressive spacing, and the preferred element size at the 15th node is 0.0070.

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
$     CPID                ICASE
      1001                    1
$                x1                  x2                  xl
                  1                  .0
                  8              0.0755              0.0070
                 15              0.1245              0.0070
                 22                 0.2
```

The above setup will produce a progressive mesh for regions 1 and 3 (indicated by 1 XL value for each region's two end points) and a uniform mesh for region 2 (indicated by 2 XL values for the region's two end points). The element size is 0.0155 at the 1st node; gradually reduces to 0.007 at the 8th node; remains unchanged until the 15th node; then gradually increased to 0.0155 at the 22nd node.

We include two other examples to illustrate the versatility of the setup. For CPID 1002, regions 1 and 3 have progressive meshes and region 2 has a uniform mesh. For CPID 1003, regions 1 and 2 both have progressive meshes.

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1002                    1
$                x1                  x2                  xl
                  1                  .0              0.0155
                  8              0.0755
                 15              0.1245
                 22                 0.2              0.0155
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1003                    1
$                x1                  x2                  xl
                  1                  .0
                 11                 0.1              0.0070
                 21                 0.2
```

5. **ICASE = 2.** For ICASE = 2, the idea is to build the progressive spaced mesh starting from a point and extend the mesh to the left and the right. To use this option, we need to first pick a "base node" and put it at a location by specifying its coordinate. The "base node" in the example below is the 8th node with a coordinate of 0.0755. Next, we specify the element size at all the nodes listed. In the example below, we want the element size to be 0.0155 at the 1st node, 0.007 at the 8th node, 0.007 at the 15th node and 0.0155 at the 22nd node.

Internally, the mesh is divided into 3 regions as discussed for ICASE = 1 in Remark 4. Progressive meshes are used between two control points with different element lengths while uniform meshes are used between two control points with the same element length. In the example below, regions 1 and 3 have progressive mesh spacing while region 2 has an evenly distributed mesh.

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
$     CPID                ICASE
      1001                    1
$                x1                  x2                  xl
```

```
                  1                                0.0155
                  8             0.0755             0.0070
                 15                                0.0070
                 22                                0.0155
```

We include two other examples to illustrate the versatility of the setup. For CPID 1002, regions 1 and 3 have progressive meshes and region 2 has a uniform mesh. For CPID 1003, regions 1 and 2 both have progressive meshes.

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1002                   1
$                 x1                   x2                   xl
                  1                                       0.0155
                  8                                       0.0070
                 15                                       0.0070
                 22                  0.2                  0.0155

*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1003                   1
$                 x1                   x2                   xl
                  1                                       0.0155
                 12                  0.1                  0.0070
                 22                                       0.0155
```

Examples:

1. The example below generates a regular box mesh. Its size is 0.2 × 0.2 × 0.2. It is generated in a local coordinate system defined by nodes 2, 3, and 4 that originates from node 1.

   The local *x*-axis mesh spacing is defined by control points ID 1001. It has 21 nodes evenly distributed from 0.0 to 0.2. The local *y*-axis is defined by ID 1002 and has twice the number of elements of 1001. It has 41 nodes evenly distributed from 0.0 to 0.2. The local *z*-axis is defined by ID 1003. It has 31 nodes and covers from 0.0 to 0.2. The mesh is two times finer in the region between node 6 and node 26.

```
*ALE_STRUCTURED_MESH
$    mshid      dpid      nbid      ebid
         1         1    200001    200001
$    cpidx     cpidy     cpidz      nid0     lcsid
      1001      1002      1003         1       234
*DEFINE_COORDINATE_NODES
$      cid      nid1      nid2      nid3
       234         2         3         4
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1001
$                  x1                   x2
                    1                   .0
                   21                   .2
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1002
$                  x1                   x2
                    1                   .0
                   41                   .2
```

```
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1003
$                 x1                  x2
                   1                  .0
                   6                 .05
                  26                 .15
                  31                  .2
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
*END
```

2. This example shows how to generate a progressively larger/smaller mesh spacing. The mesh geometry is the same as the example above. In the *x*-direction the mesh becomes progressively smaller between nodes 1 and 8. For these 7 elements, each element is 0.1/1.1 = 9.09% smaller than its left neighbor. Between nodes 15 and node 22, the elements become progressively larger by 10% in comparison to each element's left neighbor. The 7 elements in the middle between node 8 and 15 are of equal length.

```
*ALE_STRUCTURED_MESH
$    mshid      dpid      nbid      ebid
         1         1    200001    200001
$    cpidx     cpidy     cpidz      nid0     lcsid
      1001      1002      1003         1       234
*DEFINE_COORDINATE_NODES
$      cid      nid1      nid2      nid3
       234         2         3         4
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1001
$                 x1                  x2              ratio
                   1                  .0              -0.1
                   8          0.06666667
                  15          0.13333333               0.1
                  22                  .2
```

*ALE_STRUCTURED_MESH_JOIN

Purpose: Join structured ALE meshes generated by the *ALE_STRUCTURED_MESH keyword card. This keyword merges nodes along a common boundary between the specified meshes.

As many cards as needed can be input with each card representing one mesh generated with *ALE_STRUCTURED_MESH to be joined to the greater mesh. This input ends at the next keyword ("*") card.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID							
Type	I							
Default	0							

VARIABLE	DESCRIPTION
MSHID	S-ALE mesh ID for the mesh generated with *ALE_STRUCTURED_MESH

*ALE_STRUCTURED_MESH_MOTION

Purpose: Control the motion of a structured 3D ALE mesh generated by *ALE_STRUCTURED_MESH. Two motions are implemented. With the first, the mesh follows the center of mass of certain fluids and expands/contracts at the same rate as the specified fluids. With the second, the mesh follows the motion of a Lagrangian structure and expands/contracts so that the structure is covered by the mesh.

Card Summary:

Include as many of the following card types as needed. For example, you can include 1b, 1a, 1a, 1a, and 1b in a single input. This input ends at the next keyword ("*") card.

Each card represents a mesh motion operation. The OPTION field specifies the kind of S-ALE mesh motion:

- FOLLOW_GC makes the mesh follow the mass center of certain fluid(s) in the mesh and expand/contract at the same rate as those fluid(s). This mesh motion is useful, for instance, when modeling a moving projectile hitting a target.
- COVER_LAG makes the mesh follow the motion of a Lagrangian structure and expand/contract so that the Lagrangian structure is fully covered by the S-ALE mesh. This motion is useful for modeling airbag deployment.

Card 1a. Include this card when OPTION = FOLLOW_GC.

MSHID	OPTION	AMMGSID	EXPLIM				ISYM

Card 1b. Include this card when OPTION = COVER_LAG.

MSHID	OPTION	SID	STYPE	NODCEN		FRCPAD	

Data Card Definitions:

Fluid-Controlled Motion Card. Use this card when OPTION = FOLLOW_GC.

Card 1a	1	2	3	4	5	6	7	8
Variable	MSHID	OPTION	AMMGSID	EXPLIM				ISYM
Type	I	A	I	F				I
Default	none	none	0	1.0				0

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID defined in *ALE_STRUCTURED_MESH
OPTION	FOLLOW_GC
AMMGSID	Set of ALE multi-material group list IDs which the mesh follows. Please refer to *SET_MULTI-MATERIAL_-GROUP_LIST card for details.
EXPLIM	Limit ratio for mesh expansion and contraction. The distance between the nodes is not allowed to increase by more than a factor EXPLIM or decrease to less than a factor 1/EXPLIM. Default value of 1.0 means no expansion/contraction.
ISYM	A three digit number, [XYZ], to define symmetry: $$\text{ISYM} = 100 \times X + 10 \times Y + Z$$ Each digit specifies one direction, X for the local x, Y for local y, and Z for local z (local x, y, and z are defined in *ALE_STRUCTURED_MESH). Each digit can have the following values: EQ.0: No symmetry EQ.1: Symmetry plane along the face with a normal vector in the negative direction EQ.2: Symmetry plane along the face with a normal vector in the positive direction For example, ISYM = 201 means quarter symmetry with symmetry planes at the faces with normal vectors $+x$ and $-z$. 111 means 1/8 symmetry with symmetry planes at the faces with normal vectors $-x$, $-y$ and $-z$.

Structure-Controlled Motion Card. Include this card if OPTION = COVER_LAG.

Card 1b	1	2	3	4	5	6	7	8
Variable	MSHID	OPTION	SID	STYPE	NODCEN		FRCPAD	
Type	I	A	I	I	I		F	
Default	none	none	none	0	none		0.1	

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID. Defined in *ALE_STRUCTURED_MESH.
OPTION	COVER_LAG
SID	Set ID to identify the Lagrange structure
STYPE	Set type for SID: EQ.0: Part set EQ.1: Part EQ.2: Segment set EQ.3: Node set
NODCEN	Optional node ID used as the center of mesh expansion
FRCPAD	Multiplier that causes the S-ALE mesh to extend beyond the Lagrangian structure in each of the local directions of the S-ALE mesh (see *ALE_STRUCTURED_MESH). In each direction, the length of the S-ALE is calculated as: $$l_i^{\mathrm{S}} = (1 + 2 \times \mathrm{FRCPAD}) \times l_i^{\mathrm{L}}$$ where l_i^{S} is the length of the S-ALE mesh in the i^{th} direction ($i = 1, 2, 3$) and l_i^{L} is the longest length of the Lagrangian structure in the i^{th} direction. The padding beyond the structure is applied evenly to each side. FRCPAD prevents the fluid-structure interaction (FSI) from occurring on the edge of the S-ALE mesh.

*ALE_STRUCTURED_MESH_REFINE_{OPTION}

For *OPTION* the available options are:

<BLANK>

REGION

Purpose: Refine existing structured ALE (S-ALE) meshes generated by the *ALE_STRUCTURED_MESH keyword. This keyword automatically updates sets defined using the values SALECPT and SALEFAC in the OPTION field for the keywords *SET_SOLID_GENERAL, *SET_SEGMENT_GENERAL, and *SET_NODE_GENERAL. As a result, this card is the only modification needed in the input deck for refining an S-ALE mesh.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID	IFX	IFY	IFZ				
Type	I	I	I	I				
Default	none	1	1	1				

Local Refinement. Include this card for the REGION keyword option.

Card 2	1	2	3	4	5	6	7	8
Variable	IMIN	IMAX	JMIN	JMAX	KMIN	KMAX		
Type	I	I	I	I	I	I		
Default	none	none	none	none	none	none		

VARIABLE	DESCRIPTION
MSHID	S-ALE mesh ID for the S-ALE mesh to be refined
IFX, IFY, IFZ	Refinement factor for each local direction. The number of elements in each direction of the new mesh is the refinement factor for that direction multiplied by the current number of elements in that direction. They must be integers.

VARIABLE	DESCRIPTION
	IFX.LT.0: |IFX| is the level of refinement in all directions.
IMIN,IMAX, JMIN,JMAX, KMIN,KMAX	Minimum and maximum nodal indices (or control points) along each direction of the S-ALE mesh that defines a region to refine. These integers should be between the smallest and largest control points of *ALE_STRUCTURED_MESH_CONTROL_POINTS keywords called by *ALE_STRUCTURED_MESH to create the mesh with the specified MSHID. Note that for 2D S-ALE, KMIN and KMAX are ignored.

Remarks:

1. **Multi-material problems.** This keyword provides a modeling technique to handle multi-material ALE problems. In comparison to pure Lagrange problems, models containing multi-material ALE fluids are often time consuming and memory demanding. Therefore, a recommended approach to this problem is constructing a concept model with a coarse mesh to estimate the computational resources required and then refining the concept model mesh gradually until convergence is achieved. This keyword minimizes the user's effort to follow such a procedure.

Example:

The example below generates a regular evenly distributed box mesh with mesh ID 1. The S-ALE mesh has 22 nodes along each direction, and the overall size is 0.2 by 0.2 by 0.2. S-ALE mesh 1 is generated using the local coordinate system defined by nodes 2, 3, and 4 with the origin located at node 1.

```
*ALE_STRUCTURED_MESH
$     mshid      dpid      nbid      ebid
          1         1    200001    200001
$     cpidx     cpidy     cpidz      nid0     lcsid
       1001      1001      1001         1       234
*DEFINE_COORDINATE_NODES
$       cid      nid1      nid2      nid3
        234         2         3         4
*SET_SOLID_GENERAL
$       SID
        100
$    OPTION     MSHID       XMN       XMX       YMN       YMX       ZMN       ZMX
    SALECPT         1         1        22         1        22        11        22
*ALE_STRUCTURED_MESH_CONTROL_POINTS
      1001
$                 x1                  x2
```

```
                  1                  .0
                 22                  .2
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
       5   0.0000000e+00   0.0000000e+00   0.0000000e+00
*END
```

If at a later time we decided to make the mesh finer, we can simply add the following card to the input deck:

```
*ALE_STRUCTURED_MESH_REFINE
$    mshid       ifx       ify       ifz
         1         3         3         3
```

The mesh now contains 63 elements (64 nodes) in each direction. The solid element set 100 now contains elements ranging between nodes (1,1,31) and (64,64,64), instead of the original (1,1,11) and (22,22,22).

*ALE_STRUCTURED_MESH_TRIM

Purpose: Perform trim/un-trim operations on a structured ALE mesh generated by the *ALE_STRUCTURED_MESH keyword card. This keyword is supported for 2D and 3D S-ALE meshes.

As many cards as needed can be input with each card representing one mesh trim/un-trim operation. This input ends at the next keyword ("*") card. These operations are completed one by one, in the order of their cards' appearance. Please see the examples below.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID	OPTION	OPER	IOUTIN	E1	E2	E3	E4
Type	I	A	I	I	I or F	I or F	I or F	I or F
Default	0	none	0	0	none	none	none	None

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID.
OPTION	There are six available options. They are trim by: PART-SET, SEGSET, PLANE, CYLINDER, BOXCOR, BOXCPT and SPHERE. See the table below for more details.
OPER	To trim or un-trim, that is, to delete the picked elements or keep them. EQ.0: Trim (default) EQ.1: Keep
IOUTIN	Flag to select which elements to trim, that is, "outside" or "inside" the specified object defined with the OPTION and E*n*. For PARTSET and SEGSET options, "outside" is defined as the region to which the segment normal points. EQ.0: Outside (default) EQ.1: Inside
E1, E2, E3, E4	These values have different definitions for different options. See the table below for details.

The "OPTION" column in the table below enumerates the allowed values for the OPTION variable as well as describing E1, …, E4 for each OPTION. Each of the following operations accepts up to 4 arguments but may take fewer. Values of E*n* left unspecified are ignored.

OPTION	DESCRIPTION
PARTSET	Trim by PARTSET. E1 is the shell part set ID. E2 is the distance. Elements farther away than the distance in the direction of the shell normal vectors (depending on the value of IOUTIN) are deleted/kept. Please note, only elements on one side will be deleted. To delete the elements on both sides, repeat the card with the IOUTIN value reversed.
SEGSET	Trim by SEGMENT SET. E1 is the segment set ID. E2 is the distance. Elements farther away than the distance in the direction of the segment normal vectors (depending on the value of IOUTIN) are deleted/kept. To delete both sides, repeat the card with the IOUTIN value reversed.
PLANE	Trim by PLANE. E1 is the node ID of a node on the plane. E2 is another node ID off the plane. And vector E2 – E1 is normal to the plane.
CYLINDER	Trim by CYLINDER. E1, E2 are node IDs of the center nodes at two ends. E3, E4 are the radius at those two ends.
BOXCOR	Trim by BOX. E1 is the BOX ID. Please refer to *DEFINE_BOX for details on setting up a box in global coordinate system or *DEFINE_BOX_LOCAL in local coordinate system.
BOXCPT	Trim by BOX. The box is defined using S-ALE control points (CPT). E1 is BOX ID. Please refer to *DEFINE_BOX for details on setting up a box.
SPHERE	Trim by SPHERE. E1 is the node ID of the sphere center node. E2 is the radius of the sphere.

Examples:

This example shows how to trim a mesh generated by *ALE_STRUCTURED_MESH card. We use the same mesh as the example in the *ALE_STRUCTURED_MESH card manual

page, but now we trim the mesh so any elements outside of a sphere centered at (0.0,0.0,0.0) with a radius of 0.1 are deleted.

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz      nid0     lcsid
      1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$    mshid    option      oper    ioutin       nid    radius
         1    SPHERE                            1      0.10
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
*END
```

If instead we want to trim the elements inside the sphere, we simply change ioutin to 1 as follows.

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz      nid0     lcsid
      1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$    mshid    option      oper    ioutin       nid    radius
         1    SPHERE                 1         1      0.10
*END
```

If we want to trim the elements between two spheres both centered at (0.0,0.0,0.0) with radii of 0.05 and 0.1, respectively, multiple methods requiring two cards can achieve this. In one method, first delete all the elements inside the large sphere (0.1 radius). Then un-delete, the elements inside the small sphere (0.05 radius).

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz      nid0     lcsid
      1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$    mshid    option      oper    ioutin       nid    radius
         1    SPHERE                 1         1      0.10
         1    SPHERE         1       1         1      0.00
*END
```

Another method is to first delete all the elements outside the small sphere (0.05 radius) and then un-delete the elements outside the large sphere (0.1 radius).

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz      nid0     lcsid
```

```
      1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$    mshid    option      oper    ioutin       nid    radius
         1    SPHERE                 0         1      0.05
         1    SPHERE          1      0         1      0.10
*END
```

To use BOXCPT, we define a box using S-ALE control point numbers. The example below deletes all elements outside of a box with two endpoints at (8.0,8.0,8.0) and (15.0,15.0,15.0) in S-ALE control points.

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz      nid0     lcsid
      1001      1001      1001         1       234
*ALE_STRUCTURED_MESH_TRIM
$    mshid    option      oper    ioutin     boxid
         1    BOXCPT                 0         1
*DEFINE_BOX
$    boxid       xmn       xmx       ymn       ymx       zmn       zmx
         1         8        15         8        15         8        15
*END
```

*ALE_STRUCTURED_MESH_VOLUME_FILLING

Purpose: Perform volume filling operations on a structured ALE mesh generated by the *ALE_STRUCTURED_MESH keyword card.

In a typical Structured ALE (S-ALE) simulation, *ALE_STRUCTURED_MESH constructs the S-ALE mesh. The mesh includes a set of hex solid elements and their nodes. *ALE_-STRUCTURED_MULTI-MATERIAL_GROUP is used to specify the multi-materials flow in that mesh and their material properties.

With the above defined, we have the S-ALE mesh which defines our domain for the ALE simulation, and the materials flowing inside, but we have no idea of how those materials occupy the domain. We have two methods for identifying how the materials occupy the domain:

1. The first is to use *INITIAL_VOLUME_FRACTION. With this keyword, we list elements one by one and then for each AMMG in each element, its volume fraction. This approach is tedious and impractical, so it is rarely used.

2. The second way is to fill the volume automatically based on instructions given with *ALE_STRUCTURED_MESH_VOLUME_FILLING. Each instance of *ALE_STRUCTURED_MESH_VOLUME_FILLING with its data cards represents an instruction. Those instructions are executed consecutively, based on the order of appearance in the input file. Each instruction fills the volume either inside or outside of a geometry type (field GEOM in Card 2) with a certain AMMG.

Each *ALE_STRUCTURED_MESH_VOLUME_FILLING keyword has two data cards. Multiple *ALE_STRUCTURED_MESH_VOLUME_FILLING keywords can be defined and are executed in the order of appearance. Please see the examples below.

Card 1	1	2	3	4	5	6	7	8
Variable	MSHID		AMMGTO		NSAMPLE			VID
Type	I		A		I			I
Default	0		0		3			none

Card 2	1	2	3	4	5	6	7	8
Variable	GEOM	IN/OUT	E1	E2	E3	E4	E5	
Type	A	I	I or F	I or F	I or F	I or F	I or F	
Default	none	0	none	none	none	none	none	

VARIABLE	DESCRIPTION
MSHID	S-ALE Mesh ID. A unique number must be specified.
AMMGTO	AMMG name (AMMGNM on *ALE_STRUCTURED_MULTI-MATERIAL_GROUP) of the ALE multi-material group filling the geometry. See *ALE_STRUCTURED_MULTI-MATERIAL_GROUP for reference.
NSAMPLE	Number of sampling points. In case an element is partially filled, in each direction, 2 × NSAMPLE + 1 points are generated. These $(2 \times \text{NSAMPLE} + 1)^3$ points, each representing a volume, are used to determine if its volume is in or out.
VID	ID of *DEFINE_VECTOR card. This flag is used to assign initial velocity to material filling the domain. Fields 2 to 5 (XT, YT, ZT) of the *DEFINE_VECTOR card are used to define the initial translational velocities. Please refer to Example 1 below for usage.
GEOM	Geometry types. They are: PARTSET, PART, SEGSET, PLANE, CYLINDER, BOXCOR, BOXCPT and SPHERE. See the table below for more details.
IN/OUT	To fill inside or outside of the geometry. For PARTSET / PART / SEGSET options, inside is taken as in the normal direction of the container's segments (see Remark 1). EQ.0: Inside (default) EQ.1: Outside
E*i*	These values have different definitions for different options. See the table below for details.

The "GEOM" column in the table below enumerates the allowed values for the geometry variable as well as describing E1, …, E5 for each geometry type. Each of the following

operations accepts up to 5 arguments but may take fewer. Values of E*n* left unspecified are ignored.

GEOM	DESCRIPTION
ALL	Fill all volume inside the mesh. No additional variables are needed.
PARTSET	Geometry defined by PARTSET. E1 is a shell, thick shell, or solid element part set ID. E2 is the offset from the geometry (see Remark 2).
PART	Geometry defined by PART. E1 is a shell, thick shell, or solid element part ID. E2 is the offset from the geometry (see Remark 2).
SEGSET	Geometry defined by SEGSET. E1 is the segment set ID. E2 is the offset from the geometry (see Remark 2).
PLANE	Geometry defined by PLANE. E1 is the node ID of a node on the plane. E2 is another node ID off the plane. The vector going from node E1 to node E2 is the normal vector of the plane.
CYLINDER	Geometry defined by CYLINDER. E1 and E2 are node IDs of the center nodes at the two ends. E3 and E4 are the radii at those two ends.
BOXCOR	Geometry defined by BOX. E1 is the BOX ID. See *DEFINE_BOX for details on setting up a box in global coordinate system or *DEFINE_BOX_LOCAL in local coordinate system.
BOXCPT	Geometry defined by BOX. The box is defined using S-ALE control points (CPT). E1 is BOX ID. See *DEFINE_BOX.
ELLIPSOID	Geometry defined by ELLIPSOID. E1 is the node ID of the ellipsoid center node. E2, E3, and E4 are the radii along x, y, and z directions, respectively. E5 is the local coordinate system ID if a local coordinate system is used; see *DEFINE_COOR-DINATE_SYSTEM.

Remarks:

1. **Geometry specified with solid element parts.** When solid element parts specify the geometry for GEOM = PARTSET and PART, then their boundaries define the container geometry. The inside/outside direction depends on the normal vectors of the segments on the boundaries of the container.

2. **Offset direction for PART / PARTSET /SEGSET.** For GEOM = PART, PARTSET, and SEGSET, you can specify an offset distance for the fluid interface from the container's mesh. The direction of the offset depends on the normal vectors associated with the mesh of the container (see Remark 1). If the offset specified with E2 is greater than 0, then the offset is along the normal vectors from the boundary of the container. If it is less than 0, then it is offset in the negative normal direction from the boundary of the container.

3. **Files output for subsequent analyses.** This keyword causes file alevfrc.inc and salevfrc.lsda to be output at the end of the simulation. salevfrc.lsda is an lsda version of alefrc.inc. alefrc.inc can be included with *INCLUDE for subsequent analyses while salevfrc.lsda can be included with *INITIAL_VOLUME_FRACTION_LSDA. These files pre-fill the volume fractions and store the trimmed mesh which can save time for subsequent analyses.

Examples:

1. This example uses two *ALE_STRUCTURE_VOLUME_FILLING cards. The first fills all volume in a mesh with AMMG 1, outside air, with an AMMGNM "airout". The second fills AMMG 2, inside air, with an AMMGNM "airin" in a spherical domain. We use the same mesh as the example in the *ALE_STRUCTURED_MESH card manual page.

```
*ALE_STRUCTURED_MESH
$    mshid       pid      nbid      ebid
         1         1    200001    200001
$     nptx      npty      nptz
      1001      1001      1001
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1             airout
$     geom
       ALL
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1              airin
$     geom    in/out      node        rx        ry        rz     lcsid
 ELLIPSOID                          0.03                           234
*DEFINE_COORDINATE_NODES
$      cid      nid1      nid2      nid3      flag
       234         2         3         4         1
*NODE
       1   0.0000000e+00   0.0000000e+00   0.0000000e+00
       2   0.0000000e+00   0.0000000e+00   0.0000000e+00
       3   0.1000000e+00   0.0000000e+00   0.0000000e+00
       4   0.0000000e+00   0.1000000e+00   0.0000000e+00
       5   0.1000000e+00   0.1000000e+00   0.1000000e+00
*END
```

Or we could fill every element with AMMG 2 and then switch the domain outside of the sphere to AMMG 1 as follows:

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
```

```
         1             airin
$     geom
       ALL
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto            nsample                          vid
         1             airout
$     geom    in/out      node        rx        ry        rz     lcsid
 ELLIPSOID         1               0.03                           234
```

If we want to give the sphere filled in by AMMG 2 an initial velocity, simply define a *DEFINE_VECTOR card as below and assign its ID to field VID.

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto            nsample                          vid
         1             airin
$     geom    in/out      node        rx        ry        rz     lcsid
 ELLIPSOID                           0.03                           234
*DEFINE_VECTOR
$      vid        xt        yt        zt
         1      100.      -20.       0.0
```

2. In this model, we fill the whole mesh with AMMG 4 (“vacuum”) first. And then fill AMMG 1 (“water1”), AMMG 2 (“water2”) and AMMG 3 (“air”) into 3 containers, each defined by a Lagrangian shell part. 4 cards are required to do that.

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto            nsample                          vid
         1            vacuum
$     geom
       ALL
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto            nsample                          vid
         1           water1
$     geom    in/out       pid    offset
      PART               2001
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto            nsample                          vid
         1           water2
$     geom    in/out       pid    offset
      PART               2002
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto            nsample                          vid
         1              air
$     geom    in/out       pid    offset
      PART               2003
```

The above filling by PART cards assume the shell normal vectors point inward to the container. If the shell normal vectors point outwards, we need to assign the in/out value to 1. Assuming shell part 2003’s normal is pointing outwards, the card will need to be set as:

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto            nsample                          vid
         1              air
$     geom    in/out       pid    offset
      PART          1      2003
```

3. To use BOXCPT, we define a box using S-ALE control point indices.

```
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1            water1
$     geom
       ALL
*ALE_STRUCTURED_MESH_VOLUME_FILLING
$    mshid              ammgto             nsample                          vid
         1            water2
$     geom    in/out     boxid
    BOXCPT                   1
*DEFINE_BOX
$    boxid       xmn       xmx       ymn       ymx       zmn       zmx
         1         8        15         8        15         8        15
```

*ALE_STRUCTURED_MULTI-MATERIAL_GROUP_{OPTION}

Purpose: Specify the material properties of each ALE multi-material group (AMMG) used by the S-ALE solver. Each AMMG represents one unique "fluid" which flows in the S-ALE mesh.

To set up a typical Structured ALE (S-ALE) simulation, use *ALE_STRUCTURED_MESH, *ALE_STRUCTURED_MULTI-MATERIAL_GROUP, and *ALE_STRUCTURED_-MESH_VOLUME_FILLING. *ALE_STRUCTURED_MESH constructs the S-ALE mesh. The mesh includes a set of hex solid (quad for 2D) elements and their nodes. *ALE_-STRUCTURED_MULTI-MATERIAL_GROUP provides the ALE material properties. *ALE_STRUCTURED_MESH_VOLUME_FILLING fills the domain with these AMMGs.

> **NOTE:** You can use *ALE_MULTI-MATERIAL_GROUP to specify the AMMGs. However, this keyword provides a simpler interface because you specify the material and EOS here instead of needing a separate part ID. Because of this, you do not need a section definition. This keyword uses ELFORM = 11 on *SECTION_SOLID for 3D, ELFORM = 13 on *SECTION_ALE2D for 2D plane strain, and ELFORM = 14 or 15 on *SECTION_ALE2D for 2D axisymmetric. This keyword also allows you to give each AMMG a name and reference pressure (see Remarks 2 and 3).

Available options include:

<BLANK>

PLNEPS

AXISYM

No keyword option (<BLANK>) is for 3D S-ALE simulations. The element formulation is equivalent to ELFORM = 11 on *SECTION_SOLID. The PLNEPS and AXISYM options are for 2D S-ALE simulations to set up plane strain and axisymmetric element formulations. They correspond to ELFORM = 13 and 14 in *SECTION_ALE2D, respectively.

Card 1	1	2	3	4	5	6	7	8
Variable	AMMGNM	MID	EOSID					PREF
Type	A	I	I					F
Default	none	none	none					0.

VARIABLE	DESCRIPTION
AMMGNM	AMMG name. This name is required to identify the AMMG (S-ALE fluid). It is not case sensitive, but it needs to be unique. See Remark 2.
MID	Material ID
EOSID	Equation-of-state ID
PREF	Reference pressure of this AMMG. See Remark 3.

Remarks:

1. **AMMGID.** Each AMMG is automatically assigned an ID (AMMGID) according to its order of appearance in the input deck. The AMMG defined in the first line has an AMMGID = 1, the second line AMMGID = 2, and so on and so forth. This ID later could be used in other keyword cards, such as *SET_MULTI-MATERIAL_GROUP_LIST. The general ALE solver uses this convention which is inherited by the S-ALE solver.

2. **AMMG name.** Traditionally keywords refer to ALE multi-materials with their AMMGIDs. This ID is automatically assigned based on the order of appearance of each AMMG as discussed in Remark 1. This arrangement works in most cases but has its problems. For example, if you want to switch the order of the AMMGs or want to add or remove certain AMMGs, all references to those AMMGs are affected and need to be modified/corrected. Fixing those references is often forgotten, leading to errors.

   The name specified with AMMGNM on this keyword provides an alternative way to refer to AMMGs. Other ALE keyword cards, such as *SET_MULTI-MATERIAL_GROUP_LIST, accept it in place of AMMGID for S-ALE simulations. For example, suppose the third AMMG defined with *ALE_STRUCTURED_MULTI-MATERIAL_GROUP has AMMGNM = air. In the other keywords, we could either input 3 or air where it asks for AMMGID.

   We recommend using AMMGNM over AMMGID as it is more user-friendly and less error-prone. We also suggest using AMMGNM in a consistent manner, meaning using AMMGNM across all ALE keywords in the input deck. Mixed usage of AMMGNM and AMMGID does not affect the run, but LS-PrePost does not support it.

3. **PREF.** PREF allows you to provide different reference pressures for each AMMG. With this, we can handle mismatched reference pressures among different AMMGs.

Each ALE material definition (*MAT +*EOS) has an implicit reference pressure. Different AMMGs in the same model are generally required to have the same reference pressure. For example, assume we have two AMMGs, air and water. For the *EOS, we need to carefully choose E0 and V0 values so that their initial pressure matches this reference pressure, most likely atmospheric pressure.

Sometimes mismatch could happen. For example, air and high explosive may have different reference pressures. Air may have a reference pressure of 1 bar, but the high explosive defined with *EOS_JWL may have a reference pressure of 0 bar. Before PREF, this discrepancy was considered small and ignored. PREF resolves this discrepancy by allowing us to input 101325.0 as PREF for air and 0.0 as PREF for high explosive. Then, when force is evaluated for an element, this PREF pressure will be subtracted off from that of the corresponding material.

*ALE_STRUCTURED_POINT_SOURCE

Purpose: Specify an inlet gas injection boundary condition for a multiple-gas mixture inflow using a set of point source(s) in a structured ALE (S-ALE) mesh.

For airbag simulations using the S-ALE method, please refer to *AIRBAG_SALE, a wrapper card internally translated into several S-ALE keywords including *ALE_STRUCTURED_POINT_SOURCE during the keyword read-in phase.

Card 1	1	2	3	4	5	6	7	8
Variable	AMMGNM	LCVEL	LCT	NORIF				
Type	A	I	I	I				
Default	none	0	none	none				

Gas Component Mass Flow Rate Card. Include as many instantiations of this card as needed to define the mass flow rates of all the gas components.

Card 2	1	2	3	4	5	6	7	8
Variable	LCM1	LCM2	LCM3	LCM4	LCM5	LCM6	LCM7	LCM8
Type	I	I	I	I	I	I	I	I
Default	none	none	none	none	none	none	none	none

Orifice Cards. Include NORIF orifice cards; one for each orifice.

Card 3	1	2	3	4	5	6	7	8
Variable	NID*i*	VD*i*	AN*i*					
Type	I	I	F					
Default	none	none	none					

VARIABLE	DESCRIPTION
AMMGNM	ALE multi-material group (AMMG) name. This specifies the AMMG of the inlet gas. See *ALE_STRUCTURED_MULTI-MATERIAL_GROUP, specifically Remark 2 of that keyword.
LCVEL	Load curve ID for a curve giving the gas inlet velocity as a function of time. See Remark 1.
LCT	Load curve ID for a curve giving the gas inlet temperature as a function of time.
NORIF	Number of orifices
LCM*i*	Load curve ID for the mass flow rate of each gas component. See Remark 2.
NID*i*	Node ID defining the location of nozzle *i*
VD*i*	Vector ID defining of gas inflow direction at nozzle *i*
AN*i*	Area of nozzle *i*. See Remark 3.

Remarks:

1. **LCVEL.** The inlet velocity load curve is optional. If zero, a virtual tank test using the gas inflow data is performed during the initialization phase.

2. **Mass flow rate.** The gas mixture is assumed to have a uniform temperature ($\overline{T} \approx T_i$) and inlet velocity. However, the species in the mixture may each have a different inlet mass flow rate.

3. **AN*i*.** The orifice area is a mandatory input as its value is used in the following calculations:

   a) Making an estimate of the inflator volume. A virtual tank test is performed before the simulation to obtain an estimated gas velocity curve if LCVEL is not prescribed.

   b) Calculating the injected gas volume in a given time cycle. The gas injected in a cycle pushes the gases away and creates a volume for itself. That volume is calculated as the inlet velocity times orifice area times time step.

   c) Distributing the inflow gas between different nozzles. AN*i* divided by the sum of all nozzle areas scales the inflow gas flow rate to calculate the amount of gas flowing through nozzle *i*.

*ALE_SWITCH_MMG

Purpose: This card changes a fraction of an ALE multi-material-group (AMMGID) into another group. The fraction is to be specified by a *DEFINE_FUNCTION function. The function take as many arguments as there are fields specified on the cards in format 2.

Card 1	1	2	3	4	5	6	7	8
Variable	FR_MMG	TO_MMG	IDFUNC	IDSEGSET	IDSLDSET	NCYCSEG	NCYCSLD	
Type	I	I	I	I	I	I	I	
Default	none	none	none	0	0	50	50	

Variable Cards. Cards defining the function arguments. Include as many cards as necessary. This input ends at the next keyword ("*") card.

Card 2	1	2	3	4	5	6	7	8
Variable	VAR1	VAR2	VAR3	VAR4	VAR5	VAR6	VAR7	VAR8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

VARIABLE	DESCRIPTION
FR_MMG	This is the AMMG-SID before the switch. The AMMG-SID corresponds to the SID defined on a *SET_MULTI-MATERIAL_GROUP_LIST (SMMGL) card. This SID refers to one or more AMMGs. See Remark 1.
TO_MMG	This is the AMMG-SID after the switch. The AMMG-SID corresponds to the SID defined on a *SET_MULTI-MATERIAL_GROUP_LIST card. This SID refers to one or more AMMGs. See Remark 1.
IDFUNC	ID of a *DEFINE_FUNCTION function. This function determines the material fraction to be switched. See Example 1.

VARIABLE	DESCRIPTION
IDSEGSET	ID of *SEGMENT_SET that is used to pass geometric properties to the function specified by IDFUNC. *This field is optional*. LS-DYNA computes the segment center positions and normal vectors. For each ALE element, this data can be passed to the function IDFUNC for the segment that is closest to the element center. See Example 2.
IDSLDSET	The ID of a *SOLID_SET specifying which elements are affected by this particular instance of the *ALE_SWITCH_MMG keyword. *This field is optional*. If undefined, *ALE_SWITCH_MMG affects all ALE elements. The element centers are computed and can be used as variables in the function IDFUNC.
NCYCSEG	Number of cycles between each update of the segment centers and normal vectors (if a segment set is defined). For each update, a bucket sort is applied to find the closest segment to each ALE element. If the segment nodes are fully constrained, the segment centers and normal vectors are computed only one time.
NCYCSLD	Number of cycles between each update of the ALE element centers. For each update, a bucket sort is applied to find the closest segment to each ALE element. If the element nodes do not move (as with AFAC = -1 in *CONTROL_ALE), the element centers are computed exactly once.
VAR*i*	Variables used by IDFUNC which must be in the argument order (See Remark 2): EQ.0: See Remark 3. EQ.1: *xx*-stress for FR_MMG EQ.2: *yy*-stress for FR_MMG EQ.3: *zz*-stress for FR_MMG EQ.4: *xy*-stress for FR_MMG EQ.5: *yz*-stress for FR_MMG EQ.6: *zx*-stress for FR_MMG EQ.7: Plastic strain for FR_MMG EQ.8: Internal energy for FR_MMG EQ.9: Bulk viscosity for FR_MMG

VARIABLE	DESCRIPTION	
	EQ.10:	Volume from previous cycle for FR_MMG
	GE.11.and.LE.20:	Other auxiliary variables for FR_MMG
	GE.21.and.LE.40:	Auxiliary variables for TO_MMG (*xx*-stress, …)
	EQ.41:	Mass for FR_MMG
	EQ.42:	Mass for TO_MMG
	EQ.43:	Volume fraction for FR_MMG
	EQ.44:	Volume fraction for TO_MMG
	EQ.45:	Material volume for FR_MMG
	EQ.46:	Material volume for TO_MMG
	EQ.47:	Time
	EQ.48:	Cycle
	EQ.49:	*x*-position of the ALE element center
	EQ.50:	*y*-position of the ALE element center
	EQ.51:	*z*-position of the ALE element center
	EQ.52:	*x*-position of the segment center
	EQ.53:	*y*-position of the segment center
	EQ.54:	*z*-position of the segment center
	EQ.55:	*x*-component of the segment normal
	EQ.56:	*y*-component of the segment normal
	EQ.57:	*z*-component of the segment normal
	GE.58.and.LE.65:	*x*-positions of the ALE nodes
	GE.66.and.LE.69:	*x*-positions of the segment nodes
	GE.70.and.LE.77:	*y*-positions of the ALE nodes
	GE.79.and.LE.81:	*y*-positions of the segment nodes
	GE.83.and.LE.89:	*z*-positions of the ALE nodes
	GE.90.and.LE.93:	*z*-positions of the segment nodes
	GE.94.and.LE.101:	*x*-velocities of the ALE nodes
	GE.102.and.LE.105:	*x*-velocities of the segment nodes
	GE.106.and.LE.113:	*y*-velocities of the ALE nodes
	GE.114.and.LE.117:	*y*-velocities of the segment nodes

VARIABLE	DESCRIPTION
	GE.118.and.LE.125: z-velocities of the ALE nodes
	GE.126.and.LE.129: z-velocities of the segment nodes
	GE.130.and.LE.137: x-accelerations of the ALE nodes
	GE.138.and.LE.141: x-accelerations of the segment nodes
	GE.142.and.LE.149: y-accelerations of the ALE nodes
	GE.150.and.LE.153: y-accelerations of the segment nodes
	GE.154.and.LE.161: z-accelerations of the ALE nodes
	GE.162.and.LE.165: z-accelerations of the segment nodes
	GE.166.and.LE.173: Masses of the ALE nodes
	GE.174.and.LE.177: Masses of the segment nodes
	EQ.178: Position of the variable updated by the function (See Remark 4)
	EQ.179: Index of the multi-material group in the set
	EQ.180: Time step

Remarks:

1. **Mapping.** The multi-material group sets that are specified by the fields FR_MMG and TO_MMG must be of the same length. Multi-material groups are switched so that, for instance, the fourth multi-material group in the set FR_MMG is mapped to the fourth multi-material group in the set TO_MMG. (see Example 1).

2. **Variable Specification.** The variables are presented to the function IDFUNC as floating point data. The order of the arguments appearing in the *DEFINE_FUNCTION should match the order of the variables, VAR*i*, specified on Card 2 (for this keyword). For example, when there is one card in format 2 containing "47, 48, 41, 42", then the time (47), the cycle (48), and the masses (41 & 42) should be the first, second, third, and fourth arguments to the function defined on the *DEFINE_FUNCTION keyword.

   If there is a blank column between 2 variables, the list between these 2 ranks is specified. For example, if the card contains "1, ,6", then the 6 stresses (1 through 6) are selected as arguments (see Example 2). In the case that there are several groups in the sets, if a variable is repeated, the corresponding variable will be defined in the function for the same number of groups as variable instances. For instance, if the sets have 3 groups and the volume fractions of the 2 first groups

in the set TO_MMG are required as arguments of the function, a card in format 2 should have "44,44".

3. **Variable Update for Several Groups.** If there is more than one group in the set, the function is called for each group. For a given group that has an index in the set > 1 (VAR*i* = 179), some variables, including the volume fraction, mass, and internal energy, may have been updated during the previous switches. If their original values are required, they can be obtained by setting the first field (VAR1) to 0.

4. **Variable Update by the User.** The variables can be updated by the user. If VAR*i* < 0 for some variables, the function is called again (after the switch) for each of these variables. VAR*i* = 178 gives the position of the variable for which the function is called. The function's return value is taken as the new value for this variable (instead of the fraction of material to switch). If the position given by VAR*i* = 178 is zero, it means that the function is called for the switch. Only the 46 first variables 1 < VAR*i* < 46, 58 < VAR*i* < 165 and VAR*i* = 180 can be modified.

Example 1:

The first example switches the material if the pressure is lower than a given value.

```
*comment
units: mks

Switch from the 3rd group to the 5th one if the pressure of the 3rd group
is lower than pc : pres < pc
Do the same for the switch from 4th to 7th
If the switch occurs, the function frac returns 1.0. So the whole
material is permuted.

xxsig : xx-stress of the groups in the 1st *set_multi-material_group_list
yysig : yy-stress of the groups in the 1st *set_multi-material_group_list
zzsig : zz-stress of the groups in the 1st *set_multi-material_group_list
pres  : pressure
pc    : pressure cutoff
*ALE_SWITCH_MMG
$#  fr_mmg    to_mmg    idfunc  idsegset  idsldset   ncycseg   ncycsld
         1         2        10
         1         2         3
*set_multi-material_group_list
1
3,4
*set_multi-material_group_list
2
5,7
*DEFINE_FUNCTION
10
float frac(float xxsig, float yysig, float zzsig)
{
  float pc;
  pres=-(xxsig+yysig+zzsig)/3.0;
```

```
  pc=-1000;
  if (pres<pc) {
    return 1.0;
  } else {
    return 0.0;
  }
 }
```

Example 2:

The second example switches the material if it goes through a segment.

```
*comment
units: mks

Switch the 1st group to the 2nd group if the ALE element center goes
through a segment of the set defined by idsegset=1.
The segment position is updated every cycle
A fraction of the material is switched.  This fraction depends on the
distance between the segment and element centers

time    : 47th variable
cycle   : 48th variable
xsld    : 49th variable (x-position of the element center)
ysld    : 50th variable (y-position of the element center)
zsld    : 51th variable (z-position of the element center)
xseg    : 52th variable (x-position of the segment center)
yseg    : 53th variable (y-position of the segment center)
zseg    : 54th variable (z-position of the segment center)
xn      : 55th variable (x-component of the segment normal)
yn      : 56th variable (y-component of the segment normal)
zn      : 57th variable (z-component of the segment normal)
volmat1 : 43th variable (material volume of the 1st group)
volfrac1: 45th variable (volume fraction of the 1st group)
segsurf : segment surface (given by 0.5*sqrt(xn*xn+yn*yn+zn*zn))
sldvol  : ALE element volume (given by volmat1/volfrac1)
segcharaclen: characteristic length for the segment
sldcharaclen: characteristic length for the solid
xseg2sld: x-component of the vector segment center to element center
yseg2sld: y-component of the vector segment center to element center
zseg2sld: z-component of the vector segment center to element center
distnormseg2sld: Distance segment-element projected on the normal
disttangseg2sld: Distance segment-element projected on the segment plane
*ALE_SWITCH_MMG
$#  fr_mmg    to_mmg    idfunc  idsegset  idsldset   ncycseg   ncycsld
         1         2        11         1                  1
        47                  57        43        45
*set_multi-material_group_list
1
1
*set_multi-material_group_list
2
2
*DEFINE_FUNCTION
11
float switchmmg(float time, float cycle,
                float xsld, float ysld, float zsld,
                float xseg, float yseg, float zseg,
```

```
                float xn, float yn,  float zn,
                float volmat1, float volfrac1)
{
  float segsurf, sldvol, segcharaclen, sldcharaclen;
  float xseg2sld, yseg2sld, zseg2sld, distnormseg2sld;
  float xtangseg2sld, ytangseg2sld, ztangseg2sld, disttangseg2sld;
  float frac;
  segsurf=sqrt(xn*xn+yn*yn+zn*zn);
  if (segsurf != 0.0) {
    xn=xn/segsurf;
    yn=yn/segsurf;
    zn=zn/segsurf;
  }
  segsurf=0.5*segsurf;
  sldvol=volmat1/volfrac1;
  segcharaclen=0.5*sqrt(segsurf);
  sldcharaclen=0.5*sldvol**(1.0/3.0);
  xseg2sld=xsld-xseg;
  yseg2sld=ysld-yseg;
  zseg2sld=zsld-zseg;
  distnormseg2sld=xseg2sld*xn+yseg2sld*yn+zseg2sld*zn;
  xtangseg2sld=xseg2sld-distnormseg2sld*xn;
  ytangseg2sld=yseg2sld-distnormseg2sld*yn;
  ztangseg2sld=zseg2sld-distnormseg2sld*zn;
  disttangseg2sld=xtangseg2sld*xtangseg2sld+
                  ytangseg2sld*ytangseg2sld+
                  ztangseg2sld*ztangseg2sld;
  disttangseg2sld=sqrt(disttangseg2sld);
  if (disttangseg2sld <= segcharaclen &&
      distnormseg2sld <= sldcharaclen) {
    sldcharaclen=2.0*sldcharaclen;
    frac=distnormseg2sld/sldcharaclen;
    frac=0.5-frac;
    return frac;
  } else {
    return 0.0;
  }
}
```

*ALE_TANK_TEST

Purpose: Control volume airbags (*AIRBAG_) only require two engineering curves to define the gas inflator, i.e. $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$; those two curves can be experimentally measured. However, the ALE inflator needs one additional state variable - the inlet gas velocity which is impractical to obtain. This keyword provides such a curve through an engineering approximation.

It takes two curves from the accompanying *SECTION_POINT_SOURCE as input. It assumes inflator gas under choking condition to generate the velocity curve. During this process, the original curves, $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$, are modified accordingly.

This keyword complements and must be used together with the*SECTION_POINT_-SOURCE command. Please see *SECTION_POINT_SOURCE for additional information.

Card 1	1	2	3	4	5	6	7	8
Variable	MDOTLC	TANKV	PAMB	PFINAL	MACHL	VELMAX	AORIF	
Type	I	I	I	I	F	F	F	
Default	0	0.0	0.0	0.0	0.0	0.0	0.0	

Card 2	1	2	3	4	5	6	7	8
Variable	AMGIDG	AMGIDA	NUMPNT					
Type	I	I	I					
Default	0	0	50					

VARIABLE	DESCRIPTION
MDOTLC	LCID for mass flow rate as a function of time. This may be obtained directly from the control-volume type input data.
TANKV	Volume of the tank used in a tank test from which the tank pressure is measured; $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ are computed from this tank pressure data.

VARIABLE	DESCRIPTION
PAMB	The pressure inside the tank before jetting (usually 1 bar).
PFINAL	The final equilibrated pressure inside the tank from the tank test.
MACHL	A limiting MACH number for the gas at the throat (MACH = 1 preferred).
VELMAX	Maximum allowable gas velocity across the inflator orifice (not preferred).
AORIF	Total inflator orifice area (optional, only needed if the *SECTION_POINT_SOURCE card is not used).
AMGIDG	The ALE multi-material group ID (AMMGID) of the gas.
AMGIDA	The ALE multi-material group ID (AMMGID) of the air.
NUMPNT	The number of points in $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ curves. If NUMPNT = 0, defaults to 50 points.

Remarks:

In an airbag inflator tank test, the tank pressure data is measured. This pressure is used to derive $\dot{m}(t)$ and to estimate $\overline{T}_{\text{gas}}(t)$, the stagnation temperature of the inflator gas. This is done by applying a lumped-parameter method to the system of conservation equations using an equation of state.

Together $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ provide enough information to model an airbag with the control volume method (see *AIRBAG cards). However, for an ALE or Eulerian fluid-structure interaction analysis, the gas velocity, $v(t)$, and density, $\rho(t)$, at the inlet must be computed. But, since only $\dot{m}(t)$ is known, additional assumptions must be made about the inlet conditions. If $v(t)$ and $\rho(t)$ are calculated outside of LS-DYNA, then LS-DYNA combines them with $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ to obtain $\overline{T}_{\text{gas corrected}}(t)$, $v(t)$ and $\rho(t)$ which are sufficient input for an ALE calculation.

The curves $v(t)$ and $\rho(t)$ need not be calculated outside of LS-DYNA as LS-DYNA features a method for calculating them itself. This card, *ALE_TANK_TEST, activates this capability. Thus, with the combination of this card and the *SECTION_POINT_SOURCE card, LS-DYNA can proceed directly from the control volume method input, $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$, to an ALE or Eulerian fluid-structure interaction analysis. The user does not have to do the conversion himself.

If the *ALE_TANK_TEST card is present:

1. The definitions of the relative volume, $V(t)$, and the velocity, $v(t)$, curves in the *SECTION_POINT_SOURCE card will be ignored in favor of those computed by LS-DYNA.

2. The $\dot{m}(t)$curve is read in on *ALE_TANK_TEST card.

3. The $\overline{T}_{\text{gas}}(t)$ curve (stagnation temperature), as opposed to $\overline{T}_{\text{gas corrected}}(t)$, is read in on *SECTION_POINT_SOURCE card.

   There is a subtle, but important, distinction between the two temperatures. $\overline{T}_{\text{gas}}(t)$ is derived directly from the tank pressure data based on a lump-parameter approach, whereas $\overline{T}_{\text{gas corrected}}(t)$ is computed from $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$ with additional isentropic and sonic flow assumptions for the maximum velocity at an orifice. $\overline{T}_{\text{gas corrected}}(t)$ is most appropriately interpreted as the static temperature. These assumptions provide a necessary and physically reasonable supplement to the governing equation,

   $$\dot{m}(t) = \rho(t)v(t)A$$

   in which only $\dot{m}(t)$ and A are known, leaving two parameters, $\rho(t)$ and $v(t)$, as unkown.

4. The inflator area is computed from the *SECTION_POINT_SOURCE card that has the AMMGID of the inflator gas in the *ALE_TANK_TEST card. If the *BOUNDARY_AMBIENT_EOS card is used instead of the *SECTION_POINT_-SOURCE card, then the area may be input in this *ALE_TANK_TEST card.

5. The reference density of the propellant "gas," ρ_0, is computed internally and automatically used for the calculation. The ρ_0 value from the *MAT_NULL card is ignored.

Example:

Consider a tank test model that consists of the inflator gas (PID 1) and the air inside the tank (PID 2). The following information from the control volume model is available:

- $\dot{m}(t)$ (LCID 1 is from control volume model input).
- $\overline{T}_{\text{gas}}(t)$ (LCID 2 is from control volume model input).
- Volume of the tank used in the inflator tank test.
- Final equilibrated pressure inside the tank.
- Ambient pressure in the air.

Also available are:

- The nodal IDs of the nodes defining the orifice holes through which the gas flows into the tank.
- The area associated with each hole (the node is assumed to be at the center of this area).
- The vector associated with each hole defining the direction of flow.

In the input below LCID 1 and 2 are $\dot{m}(t)$ and $\overline{T}_{\text{gas}}(t)$, respectively. LCID 4 and 5 will be ignored when the *ALE_TANK_TEST card is present. If it is not present, all 3 curves in the *SECTION_POINT_SOURCE card will be used. When the *SECTION_POINT_-SOURCE card is present, the element formulation is equivalent to an ELFORM = 11.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
*PART
inflator gas
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         1         1         1         0         0         0         0         0
*PART
air inside the tank
$      PID     SECID       MID     EOSID      HGID      GRAV    ADPOPT      TMID
         2         2         2         0         0         0         0         0
*SECTION_SOLID
$    SECID    ELFORM       AET
         2        11         0
*ALE_MULTI-MATERIAL_GROUP
$      SID   SIDTYPE
         1         1
         2         1
*SECTION_POINT_SOURCE
$    SECID     LCIDT  LCIDVOLR   LCIDVEL      <= 3 curves in tempvolrvel.k file
         1         2         4         5
$   NODEID    VECTID      AREA
     24485         3    15.066
       ...
     24557         3    15.066
*ALE_TANK_TEST
$   MDOTLC     TANKV      PAMB    PFINAL     MACHL    VELMAX     AORIF
         1     6.0E7    1.0E-4  5.288E-4       1.0       0.0
$   AMGIDG    AMGIDA    NUMPNT
         1         2        80
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|...8
```

*ALE_UP_SWITCH

Purpose: For the simulation of airbag inflation process, this card allows the switching from an ALE or S-ALE computation to a control volume (CV) or uniform pressure (UP) method at a user-defined switch time.

Card 1	1	2	3	4	5	6	7	8
Variable	UPID	SWTIME						
Type	I	F						
Default	0	10^{16}						

Card 2	1	2	3	4	5	6	7	8
Variable	FSI_ID1	FSI_ID2	FSI_ID3	FSI_ID4	FSI_ID5	FSI_ID6	FSI_ID7	FSI_ID8
Type	I	I	I	I	I	I	I	I
Default	0	0	0	0	0	0	0	0

Hybrid Card. Include this card if UPID = 0.

Card 3	1	2	3	4	5	6	7	8
Variable	SID	SIDTYPE	MMGAIR	MMGGAS				
Type	I	I	I	I				
Default	0	0	0	0				

VARIABLE	DESCRIPTION
UPID	ID corresponding to an instantiation of *AIRBAG_HYBRID_ID card for use in an ALE or S-ALE method switching to CV method simulation. The simulation starts with the ALE or S-ALE computational method and then switches to a CV (or UP) method at some given time.

VARIABLE	DESCRIPTION
	EQ.0: The code constructs an equivalent *AIRBAG_HYBRID_ID card automatically internally (default). Card 3 is then a required input. NE.0: An ID points to a corresponding *AIRBAG_HYBRID_ID card which must be defined for use after the switch. If UPID is defined, do not define Card 3.
SWTIME	The time at which the computation does a switch from an ALE or S-ALE method to CV method.
FSI_ID1, …, FSI_ID8	Coupling IDs for one or more ALE fluid-structure-interaction (FSI) *CONSTRAINED_LAGRANGE_IN_SOLID_ID cards or S-ALE FSI *ALE_STRUCTURED_FSI_TITLE cards. These couplings are deleted during the CV computational phase.
SID	A set ID that defines the Lagrangian parts which make up the airbag
SIDTYPE	Set ID type for the above SETID (following the conventions in *AIRBAG_HYBRID card). EQ.0: SID is a segment set ID. NE.0: SID is a part set ID.
MMGAIR	The AMMG (ALE multi-material group) ID of the surrounding air
MMGGAS	The AMMG ID of inflator gas injected into the airbag

Example 1:

Consider an airbag model with a 2-phase simulation: an ALE calculation being switched to a CV method. During the CV phase, the simulation is defined by an *AIRBAG_HYBRID_ID card.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_UP_SWITCH
$     UP_ID    SW_time
     100000     2.0000
$ FSI_ID_1  FSI_ID_2  FSI_ID_3  FSI_ID_4  FSI_ID_5  FSI_ID_6  FSI_ID_7  FSI_ID_8
          1         2
$--------------------------------------------------------------------------------
*AIRBAG_HYBRID_ID
$       ID
     100000
$      SID     SIDTYP      RBID      VSCA      PSCA      VINI       MWD      SPSF
         2          1         0       1.0       1.0       0.0       0.0       0.0
$ 2    ATMT      ATMP      ATMD        GC        CC
      293. 1.0130e-4  1.200E-9    8.3143        1.
```

```
$       C23     LCC23       A23     LCA23      CP23     LCP23      AP23    LCAP23

$       OPT     PVENT      NGAS
                                4
$bac LCIDM      LCIDT   NOTUSED        MW     INITM         A         B         C
      1001       1002           0.0288691       1.0     28.98
$    FMASS

$air LCIDM      LCIDT   NOTUSED        MW     INITM         A         B         C
      1600       1603            28.97E-3       0.0     26.38  8.178e-3 -1.612e-6
$    FMASS

$pyroLCIDM      LCIDT   NOTUSED        MW     INITM         A         B         C
      1601       1603            43.45E-3       0.0     32.87  2.127e-2 -5.193E-6
$    FMASS

$sto_LCIDM      LCIDT   NOTUSED        MW     INITM         A         B         C
      1602       1603            39.49E-3       0.0     22.41  2.865e-3 -6.995e-7
$    FMASS
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```

Example 2:

Consider the same airbag model with the same 2-phase simulation. However, all the *AIRBAG_HYBRID_ID card definitions are extracted automatically from the ALE model. There is no need to define the *AIRBAG_HYBRID_ID card. Card 3 is required.

```
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
*ALE_UP_SWITCH
$    UP_ID   SW_time
$   100000    2.0000
          0    2.0000
$ FSI_ID_1  FSI_ID_2  FSI_ID_3  FSI_ID_4  FSI_ID_5  FSI_ID_6  FSI_ID_7  FSI_ID_8
          1         2
$    SETID    SETYPE   MMG_AIR   MMG_GAS
          2         1         2         1
$...|....1....|....2....|....3....|....4....|....5....|....6....|....7....|....8
```