INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI[®]

FEM MODELING OF THE DIE CASTING FILLING PROCESS

by

Guoming Li

B.S. Civil, Qingdao Institute of Architecture and Engineering, China

A thesis submitted to

the Faculty of Graduate Studies and Research

in partial fulfillment of the requirements for the degree of

Master of Engineering

Department of Mechanical and Aerospace Engineering

Ottawa - Carleton Institute for Mechanical and Aerospace Engineering

Carleton University

Ottawa, Canada

Sept, 2000

© Copyright, 2000

Guoming Li

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-57730-9

Canada

The undersigned recommend to the Faculty of Graduate Studies
and Research acceptance of the thesis

FEM MODELING OF THE DIE CASTING FILLING PROCESS

submitted by

Guoming Li

in partial fulfilment of the requirements for

the degree of

Master of Engineering

A handwritten signature in cursive script, reading "John Goldak", written over a horizontal line.

Thesis Supervisor

A handwritten signature in cursive script, reading "Robert Bell", written over a horizontal line.

Chair, Department of Mechanical and Aerospace Engineering
Carleton University

Acknowledgments

I would show my deepest appreciation for Professor John Goldak. His endless dedication, enthusiasm and patience have taught me the way to do research work and brought me to explore the die casting world. I also thank him for his consistent support during my master's study and my work. I would thank Mr. Dan Downey for providing me the insight into Linux system and MRCO utilities. Ms. Mihaela Mocanita for explaining the TCL codes from the very beginning. Jianguo and Shaodong for their help in developing the filling solver. I would give special thanks for Dave, Oly and John in H.E.Vannatter's for their invaluable assistance in my daily work. I also thank those people who have directly or indirectly contributed to this thesis and I failed to mention.

Abstract

As a contribution to develop a foundry simulation software, the algorithm and numerical results of different filling solvers are presented and compared in this thesis. The predicted flow pattern of PlicIf solver basically matches well with the actual flow in simple geometry castings. But in complex geometry problems, the liquid front face tends to follow the momentum of the air. As an improvement to the PlicIf solver, a droplet solver was developed. This is a Lagrangian solver. The metal liquid is modeled as numerous droplets. They are initialized with proper velocity and diameter at the inlet of the filling domain. The surface layer of elements on the inner side of the die wall are treated as bouncing elements. The normals for bouncing elements are obtained from the STL files. While traveling in the cavity, the droplets bounce between die walls. After travelling for a certain time interval, the droplet solver is halted and droplet data (mass and momentum) are mapped to the element data (vof and velocity). DFP (Divergence Free Projection) solver is then solved to get a better velocity field to help the droplets travel around corners. However, even with the updated velocity field from the DFP solver, the piling up of droplets at corners could not be avoided completely. As a result, after each step of filling, extra liquid in an element is mapped to their neighbouring elements to enforce incompressibility condition. A stress solver will be included later to deal with this problem more efficiently.

Nomenclature

B	Extensive property of a fluid.
$\overline{B}$	The local volume average of B in droplet flow.
C_D	The drag coefficient.
d	Diameter of a particle.
E_k	Kinetic energy.
E_i	Internal energy.
e	Sum of specific energy.
F	Force.
f	Body force.
I	Identity matrix.
g	Dirichlet data.
h	Neumann data.
h_s	The enthalpy of the fluid crossing the control surface.
$\bar{i}$	The average specific internal energy of a particle.
J	Thermal radiosity.
M	Mass.
n	Unit normal vector.
P, p	Pressure.
P_S	Percent solid fraction allowable in the metal at the end of filling.

$\dot{Q}$	The change rate of heat flux.
R	The universal gas constant.
Re	Reynolds number.
R_1, R_2	Principal radii of curvature.
r	Radius of a circle or sphere.
$\dot{r}$	The regression rate of the particle surface.
S	Surface area.
T	Temperature.
T_C	Casting thickness.
t	Time.
u	Velocity in x direction.
V	Volume.
V_r	The relative velocity of the field with respect to the particle.
v	Velocity in y direction.
v_f	Air field velocity.
v_i	The velocity of fluid with respect to inertial reference frame.
$\dot{W}$	The change rate of work.
w_i	The velocity of fluid with respect to control surface.
w_j	Rotation vector of a particle.
w'	The sum of the regression rate and efflux velocity ($\dot{r} + \omega$).

α	Thermal diffusivity.
α_1	Thermal absorptivity.
β	Intensive property of a fluid.
Γ	Boundary of the domain.
Γ_D	Dirichlet boundary.
Γ_N	Neumann boundary.
τ	Shear stress.
λ	The second coefficient of viscosity.
μ	Dynamic viscosity.
ρ	Density.
Ω	A domain in 3D or 2D.
Π	Free surface between air and metal liquid.
σ	Normal stress.
σ_1	The Stephan-Boltzmann constant.
ξ_k	The distance from the center of mass.
δ	The surface tension.
ϵ	Emissivity.
Δ	The coordinate-wise Laplacian.

Contents

Acceptance Sheet	ii
Acknowledgements	iii
Abstract	iv
Nomenclature	v
Table of Contents	viii
List of Figures	xii
List of Tables	xiv
1 Introduction	1
1.1 Casting : history and development	1
1.2 Metal-shaping methods	2
1.3 The technology of a foundry	4
1.3.1 Different casting methods	5
1.3.2 Casting quality control	6

1.3.3	Research work	8
1.4	Die casting	9
1.4.1	Types of dies	9
1.4.2	Die casting processes	9
1.4.3	Die terminology	12
1.4.4	Merits and shortcomings of die casting	13
1.4.5	Die casting design	13
1.5	Virtual die casting environment	19
2	Basic Physical and Mathematical Equations	21
2.1	The conservation equations	21
2.1.1	Conservation of mass	22
2.1.2	Conservation of momentum	22
2.1.3	Conservation of energy	26
2.2	Fluid flow equation	26
2.3	Free surface equation	28
2.3.1	Surface tension	28
2.3.2	Numerical simulation	30
2.4	Droplet equations	31
2.4.1	Reynolds transport theorem	31
2.4.2	Single particle equation	34

2.4.3	Volume averaged droplet equation	42
2.5	Coupled differential equations for casting analysis	45
3	Droplet Filling Solver	47
3.1	A review of a particle method	47
3.2	Single droplet test	48
3.2.1	The movement equations	49
3.2.2	Choosing parameters	51
3.2.3	Test results	52
4	Portrait of Solvers	57
4.1	Droplet filling solver	57
4.1.1	Determine inletDroplet number and size	57
4.1.2	Determine startTime & endTime	58
4.1.3	Making the bounce element layer	58
4.1.4	Droplet state parameters	59
4.1.5	Droplet initial parameters	59
4.1.6	Advect inlet droplets	59
4.1.7	Output from inlet droplet solver	60
4.1.8	Map results	60
4.1.9	Distribute extra vof	61

4.1.10	Make new droplets from vof	62
4.1.11	Advect vofDroplet	63
4.1.12	Different versions of droplet filling solvers	63
4.2	PlicIf solver	64
4.2.1	Boundary conditions for PlicIf	64
4.2.2	PlicIf domain	65
4.2.3	Ramp up velocity	65
4.2.4	Determine the PlicIf time interval	66
4.3	Air exhaust	66
4.3.1	Pressure boundary conditions	66
4.3.2	Cavity air	68
4.4	Cavity air and trapped air	68
4.4.1	Initialize cavity air	69
4.4.2	Find separated trapped air	69
4.4.3	Get relations and merge trapped air	69
4.4.4	Update cavity air	70
4.4.5	Update parent list	72
4.4.6	Trapped air cavities	72
4.5	Restart the job	75
4.6	How to run a filling analysis	76

4.6.1	STL files	76
4.6.2	Design file	77
4.6.3	Meshing - make grid file	82
4.6.4	Different filling solvers	82
4.6.5	Boundary conditions and initial conditions	83
4.6.6	Material properties	83
5	Numerical Results	84
5.1	Transmission test	85
5.1.1	PlicIf with changing pressure B.C.	85
5.1.2	Droplet solver	86
5.1.3	Droplet solver with DFP solving	86
5.2	Bearing plate test	87
5.2.1	Practical experiment	87
5.2.2	Numerical results from the solvers	88
5.2.3	Prediction of trapped air	88
6	Conclusion and Future Work	108
6.1	Conclusions	108
6.2	Future work	109

List of Figures

1.1	Hot chamber process	10
1.2	Cold chamber process	11
2.1	Control volume in Cartesian coordinates	23
2.2	Stresses acting on a differential volume	24
2.3	The free surface and the surface tension	29
2.4	System moving through a control volume at two successive time . . .	32
2.5	Moving control surface enclosing particle	35
2.6	Definition of surfaces	44
4.1	Data mapping from donor to acceptor elements	74
5.1	Different Perspectives of A Transmission (STL files)	89
5.2	PlicIf solver with Changing Pressure Boundary Conditions I	90
5.3	PlicIf solver with Changing Pressure Boundary Conditions II	91
5.4	Droplet Solver I	92

5.5	Droplet Solver II	93
5.6	Droplet DFP solver I	94
5.7	Droplet DFP solver II	95
5.8	Different Perspectives of Experiment Specimen	96
5.9	Front View of Experiment Specimen	97
5.10	Back View of Experiment Specimen	98
5.11	Comparison I – PlicIf Solver	99
5.12	Comparison II – PlicIf Solver	100
5.13	Comparison III – PlicIf Solver	101
5.14	Comparison IV – PlicIf Solver	102
5.15	Comparison I – Droplet DFP Solver	103
5.16	Comparison II – Droplet DFP Solver	104
5.17	Comparison III – Droplet DFP Solver	105
5.18	Comparison IV – Droplet DFP Solver	106
5.19	Prediction of Trapped Air in Casting	107

List of Tables

1.1	Comparison of different casting methods	14
3.1	Test I – droplet movement with zero relative velocity	53
3.2	Test II – droplet velocity being decelerated by the air	54
3.3	Test III – droplet velocity being accelerated by the air	55
3.4	Test IV – droplet moving with 45° inlet angle	56

Chapter 1

Introduction

1.1 Casting : history and development

Metal founding is one of the oldest of all industries. Both ancient and medieval history offer examples of the manufacture and use of castings. Among the early scientists, the work of Reaumur (1683 - 1757) is of outstanding interest to founders. Despite the interest of a few early scientists, metal casting was looked upon as an art and this view persisted until well into the present century.

Many foundries sprang up after the industrial revolution, the vast majority being for the manufacture of the cast iron then being used as a structural material. A feature of the development of the modern industry has been the shift in the responsibilities carried by operators and supervisors alike. The further development of the industry

and particularly the advent of mass production of many types of casting has brought its specialization.

The structure of the modern foundry industry is complex. Directly related to the traditional industry are the jobbing foundries with their capacity for undertaking work involving a wide range of sizes and designs. The jobbing foundry is constantly presented with new problems in the moulding of individual design features and in the determination of casting methods which will ensure a sound product at the first attempt. While design variations can be accommodated by recourse to the skill of the moulder.

In the mass production foundry, by contrast, the emphasis is upon close process control to maintain consistency in materials and procedures. Sophisticated pattern equipment eliminates the need for a high degree of skill in moulding. While there is opportunity for progressive development of the casting method to reduce margins and achieve the most economic production.

1.2 Metal-shaping methods

The principal methods of shaping metal may be classified in five groups:

1. Casting

The production of shaped articles by pouring molten metal into moulds.

2. Mechanical working

The shaping of metals in the solid state by plastic or cold working. The starting point for this group of processes is the casting ingot or billet and the metal must possess the capacity for plastic deformation. Much output in this category is of standard primary or semi-finished shapes.

3. Fabrication by joining

The production of structural units by the joining of smaller components manufactured in other ways. The most notable method employed is welding, much of which is carried out using components cut from standard wrought materials.

4. Machining

The production of shaped articles by cutting from plain or roughly shaped forms using machine tools. Machining is also often frequently used as a finishing operation to develop accurate final dimensions on components formed by other methods.

5. Powder metallurgy

The production of shaped parts by the die pressing and sintering of metal powders.

Consideration of this brief outline shows that the main shaping processes are in many respects complementary. Over wide fields, however, their capabilities overlap,

allowing alternative decisions and providing a basis for competition with respect to properties and cost.

Casting, the production of the metal founding industry, are manufactured in a single step from liquid metal without intermediate operations of mechanical working such as rolling and forging. Shaped casting are thus distinguished from ingot and other cast forms which are only at an intermediate stage of their metallurgical life.

Casting must variously compete with forgings, weldments and sintered compacts, quite apart from plastics, ceramics and other non-metallic materials. The alternative products can offer a challenge to the founder in terms both price and functional performance. Castings on the other hand, can make inroads in fields traditionally supplied by other types of product, as in the replacement of certain sheet metal constructions by pressure die castings or of machined components by investment castings.

1.3 The technology of a foundry

The processes employed in the industry include gravity and pressure die casting, investment casting and centrifugal casting, etc. The variety of technological problems encountered in metal casting arises largely from differences in the physical properties and constitution of the alloys. The most distinguished is the wide range of process temperatures for different alloys. Other significant differences include chemical ac-

tivity and solubility of gases, mode of solidification, and contraction characteristics. These affect the problems of metal flow, mould stability, feeding, stress-free cooling and the structure and properties of the cast material. Individual techniques are thus required to meet conditions peculiar to the type of the alloy.

1.3.1 Different casting methods

Die casting

Molten metal is forced into the die cavity at pressures ranging from 0.7 to 700 MPa. When the metal solidifies, the die halves open and the casting is ejected.

Sand casting

Sand casting is one of the older techniques. In this form, a mold is made from sand, and the part is cast into it. When the metal has hardened and cooled, the sand is removed.

Investment casting

Make a wax or plastic replica of the piece to be cast first. These replica are surrounded with investment material, baked out, and metal is poured into the resultant cavity. Molds are broken to remove the castings.

Shell-mold casting

Resin-coated sand is poured onto hot metal patterns, curing into shell-like mold halves. Then they are removed from the pattern and assembled with or without cores. Metal is poured into the resultant cavities. Molds are broken to remove castings.

Other casting techniques

Other casting methods include vacuum casting, permanent mold casting, centrifugal casting, slip casting, etc. The interested reader could refer to related foundry books.

1.3.2 Casting quality control

Once the essential foundry technique has been established, the quality of a casting is influenced by many process variables. Some of these can only be adjusted within narrow limits. Metal composition, for example, is often restricted by specification. Inoculation with very small amounts of certain elements can exercise potent effects both on foundry characteristics and casting structure. Meanwhile, control can also be exercised by moulding material properties, melting practice, casting temperature and pouring speed. These variables, together with the techniques of gating, risering and chilling, can be used to control the entire pattern of cooling, which is the main

key to the structure and quality of the finished casting.

Defects in castings and inspection methods

The defects that must be controlled to achieve quality can be categorized as:

1. Filling defects: Cold laps, cold shuts, excessive turbulence.
2. Inclusion defects: Slag, sand erosion, oxides.
3. Gas defects: Negative pressure, ingestion of air into castings, surface gas defects.
4. Shrinkage/Porosity defects: Defects due to volumetric contractions in solidification.
5. Cold cracks: Defects due to volumetric contractions in the solid state.
6. Dimensional errors: Distortion.

Inspection methods

Typical inspection methods use:

- polishers and microscopes to look at microscopic structures
- metal analyzer to determine chemical composition
- X-rays are used to detect hidden cracks and blowholes
- Dimensional validation using range sensors

1.3.3 Research work

The research into the fundamentals of casting and cast materials involves basic studies of the phenomena of liquid metal flow and solidification, high temperature stress-strain relationships in cast alloys and mould materials, and the origins of various forms of defects. Such research will have a major bearing on the future development of metal founding.

Perhaps the most critical areas for investigation are the process-structure-properties relationships in cast materials. Modern engineering requirements involving high stresses and extreme service conditions must be met by more exacting standards of quality and reliability. Apart from the development of new casting alloys and improved techniques of non-destructive examination, the capacity to control structure can provide more positive correlation between the true properties within castings and the nominal values. This approach is seen in the concept of 'premium' castings with their high and guaranteed levels of internal properties. It seems that such an approach will eventually form the normal basis for casting design.

1.4 Die casting

1.4.1 Types of dies

Dies are classified as: single cavity, multiple cavity, combination and unit dies. Multiple cavity dies have several cavities which are all identical. If a die has cavities of different shapes, it is called a combination or family die. A combination die is used to produce several parts for an assembly. For simple parts, unit dies might be used to affect tooling and production economies. Several parts for an assembly, or for different customers, might be cast at the same time with unit dies. One or more unit dies are assembled in a common holder and connected by runners to a common opening or sprue hole. This permits simultaneous filling of all cavities.

1.4.2 Die casting processes

Die casting can be described as the filling of a metal die with molten metal at a high velocity. The source of the power for filling comes from the die casting machine. The die casting machine is hydraulically powered. The die absorbs the stresses of injection and dissipates the heat of the metal until the metal solidifies. The part is then removed from the die. There are a variety of die casting processes. However they can be lumped into three major categories, the hot chamber, cold chamber and direct injection processes.

The hot chamber process

Hot chamber die-casting is the original system and it is still used today with low melting temperature metals such as zinc, lead, tin and magnesium alloys. A hot chamber die casting machine is shown in Figure 1.1.

Figure 1.1: Hot chamber process

This process is beneficial because it minimizes turbulence, oxidization, and heat loss during filling. However it has one significant drawback. The hydraulic actuator is in direct contact with the molten metal for long periods of time. This causes large thermal stresses on the machine and its components. It is for this reason alone that this process is not practical for higher melting point alloys. The hot chamber die casting process works as follows, the hydraulic system drives the piston downward, the plunger (submerged into the bath of molten aluminum) forces the molten metal

through the goose-neck and into the die, once in the die the metal is allowed to solidify, after solidification, the die is opened and the casting is ejected.

The cold chamber process

The cold chamber process is used for high melting point alloys such as aluminum, copper, and even some ferrous metals. The cold chamber process eliminates the problem of having the hydraulic actuator in contact with the molten metal for long periods of time. The metal must be measured for each individual casting cycle and poured into the shot sleeve immediately before the cycle begins. A cold chamber machine is shown in Figure 1.2.

Figure 1.2: Cold chamber process

After the right amount of metal is poured into the shot sleeve, the hydraulic system forces the plunger to push the liquid into the runner gate system. The flow

atomizes at the gate and fills the die cavities. Once in the die cavity, the process is exactly the same as the hot chamber process.

Direct inject

In direct injection, the technology used for polymers is extended to metals. It involves the intimacy of the hot chamber and uses many small nozzles connected to a manifold, eliminating the runner gate system.

1.4.3 Die terminology

Sprue holes are tapered with the small end located at the breaking point when the die is opened. A sprue pin, located in the ejector half, makes the sprue hollow and deflects metal entering the die into the runner system. Runners are channels located at the parting line to route liquid metal from the sprue hole to the gate.

Gates are passages through which metal enters the die cavity. They have an important function in directing metal flow so that the cavity is correctly filled. Air is expelled through vents as molten metal enters the die cavity.

Guide pins assure proper alignment of die halves and correctly register the cavities.

Side walls and cores are designed to have a slight taper or draft. The largest diameter or cross section of a cavity must be located at the parting line (unless slides are used) so the casting can be removed from the die. As molten metal in a die cavity

solidifies, it shrinks away from walls onto core pins and other projections. The design of the die must permit withdrawal of core pins, and ejection without applying too much pressure which could cause deformation to the die casting.

Cores, fixed or movable, as well as “loose pieces”, must be positioned to facilitate removal either mechanically or by hand. Loose pieces used to form undercuts must be positioned by the operator. This requires extra labor, and usually slows the casting cycle.

Inserts can be cast integrally to provide special characteristics.

Most dies are cooled by water circulating through channels drilled for that purpose.

1.4.4 Merits and shortcomings of die casting

Die casting is an efficient, economical process which, when used to its maximum potential, can replace assemblies of a variety of parts produced by various manufacturing processes at significant savings in cost and labor, Table 1.1.

1.4.5 Die casting design

There is no industry standard for the design of the die casting runner gate system. In fact the design process can vary quite significantly depending on the application of the final casting, the background of the designer and culture of the foundry. However, there are some basic steps that almost all designers follow in some form or another.

PROCESS	ADVANTAGES	LIMITATIONS
Die Casting	Excellent dimensional accuracy; Excellent surface finish; High production rate.	Die cost is high; part size limited; usually limited to non-ferrous metals; long lead time.
Investment Casting	Intricate shapes; excellent surface finish and accuracy; almost any metal cast.	Part size limited; expensive patterns, mold, and labor.
Sand Casting	Almost any metal is cast; no limit to size, shape or weight; low tooling cost.	Some finishing required; somewhat coarse finish; wide tolerances.
Shell Casting	Good dimensional accuracy and surface finish.	Part size limited; expensive patterns and equipment required.

Table 1.1: Comparison of different casting methods

These steps are as follows: Plan the metal flow paths through the cavity, compute the filling time, select the gate velocity, establish minimum and maximum gate depth, determine the gate size and shot plunger size that best matches the power required by the gating system to the power curve of the machine's inject system, make the final selection of plunger size and speed, calculate all the gate and runner dimensions [1]. The steps are briefly covered below:

Planning flow patterns

The initial stage in gating design is to plan the metal flow paths. This stage sets the design path for the rest of the design. If an error is made in this step, the rest of the design will be difficult, even impossible. There are a few guidelines set out to assist the engineer in this process:

1. The molten metal should flow in the direction of the shortest dimension in the cavity.
2. The flow paths should be parallel or diverge from each other.
3. If the plan is to disperse shrinkage evenly, then the metal should enter at the thin sections, then feed the thicker regions. If the plan is to feed shrinkage using the machine pressure, then it is important to enter the casting at the thick sections and feed across to the thinner regions.
4. When molten metal must travel a long distance in the die, it should travel in a runner and then enter the die through a gate. The flow distance through a thin cavity should be minimized.
5. All parts of all cavities in the die should become filled at the same instant.

The planned flow patterns are rarely exactly the pattern that will exist in even the best designed system [2].

Computing filling time

Computing the time needed to fill the cavity is the second step in the design process.

The filling time can be computed using the gating equation:

$$t = k \cdot ((T_1 - T_2 + P_S \cdot Z)/(T_1 - T_3)) \cdot T_C \quad (1.1)$$

where:

t = The ideal filling time (sec)

k = Empirically derived constant (sec/in)

T_1 = Temperature of the molten metal as it enters the die (deg F)

T_2 = Minimum flow temperature (deg F)

T_3 = Temperature of the die cavity surface just before the metal hits it (F)

P_S = Solid fraction percent allowable in the metal at the end of filling [typically below 0.3 solid fraction]

Z = Units conversion factor (deg F/%)

T_C = Casting thickness (in)

Selecting gate velocity

The third step is to choose a gate velocity. This is the velocity at which the molten metal moves through the gate into the cavity. There is one main parameter that affects the gate velocity. That is atomized jet flow. It is generally accepted that one of the objectives of the gate system is to atomize the flow as it enters the cavity.

Because some experimental evidence suggests that atomized flow has a much lower thermal fluxes to the die. Thus the risk of cold shuts is reduced. As a general rule, the gate velocity should be 30 to 60 m/s for aluminum alloys [2]. For ferrous metals a larger velocity is needed.

Establishing gate depth

The gate depth must satisfy the condition of flow atomization and of the requirements for trimming and finishing the final casting. The minimum gate depth can be found using the following relationship [2]:

$$\rho \times G_d \times G_v^{1.74} = J \quad (1.2)$$

where:

ρ = The density of the alloy

G_d = The gate depth (*in*)

G_v = The gate velocity (*in/sec*)

J = Derived constant for the alloy (*lb/in²*)

Maximum gate depth is governed by the thickness of the casting. A frequently used rule of thumb is that the maximum gate depth should be about 80% [2] of the adjacent casting thickness.

Balancing gate size and machine power

The fifth step in the design of the runner gate system is to match the die requirements with machine capabilities. The principal tool for this is the $P - Q^2$ diagram. The interested reader should refer to [3].

Determine the final plunger size

The sixth step in the design process is to select the appropriate plunger diameter. This step is influenced by Step 5. There are other considerations when considering plunger size. They are die flashing, wear and tear of the die, the amount of air entrapped in the molten metal and pressure feeding shrinkage and machine size constraints.

Generally, there are many physical phenomena in producing a casting. Realistic modeling of the underlying physics of casting events is fundamental to provide a scientific basis for casting design. It is possible to deal with some of these phenomena with the current level of understanding. But it is challenging in many aspects. For example, casting physics involves the interactions of a multitude of physical phenomena. It is very rare that a packaged analysis system addresses these interactions.

1.5 Virtual die casting environment

A casting analysis environment has been developed. The environment is modular which means that new or improved analysis capabilities can be added as they are developed. Coupled analysis involving interactions between several physical phenomena can be performed. In this software, parameterized features such as sprues, gates and risers can be instantiated and attached interactively to a casting and their dimensions can be manipulated. A mesh is automatically created from the designer's specifications. The current analysis capability consists of a FEM based fluid flow analysis, a thermal analysis and a stress analysis. Visualization software is included for presentation of the results. The designer is provided with a graphical user interface. The process of translating a casting problem into a problem that can be solved by FEM analysis is completely hidden from the user.

The virtual die casting environment is a junior partner of the designer in the design process. It is intelligent because it can collaborate with the designer by making some decisions. The intent is that the design assistant assumes responsibility for routine details of the design process, allowing the designer to concentrate on more productive and creative tasks.

The aid provided by this environment differs from the aid a traditional off-the-shelf analysis program would provide. Generally, in such a program, designers must explicitly specify the type of equation to solve, control all aspects of the analysis

including translation of the proposed design into initial values and boundary conditions, material properties, preparing a mesh, selecting numerical method tolerances, time step size, etc., without any help from the analysis software.

Chapter 2

Basic Physical and Mathematical Equations

In this chapter, the underlying physical and mathematical equations for FEM treatment of fluid flow will be presented. These equations are only briefly covered here instead of an exclusive account or rigorous derivation. This chapter is composed of sections containing the fluid flow equation, free surface equation, droplet equation and how they are coupled.

2.1 The conservation equations

The equations of fluid flow are derived from three conservation laws: The conservation of mass, the conservation of momentum and the conservation of energy.

2.1.1 Conservation of mass

This law means that mass can neither be created nor destroyed. This can be stated mathematically in more than one way. Given a region D , and a sub region Ω contained in D , i.e., $\Omega \subset D$. Let $\partial\Omega$ be the boundary of the region Ω , $\vec{n}$ denotes the unit outward normal. The domain Ω is fixed in space, i.e., it does not change with time. Using an Eulerian frame $(\vec{x}, t)$, the rate of change of mass in this volume can be expressed as:

$$\frac{d}{dt}m(\vec{x}, t) = \frac{d}{dt} \int_{\Omega} \rho(\vec{x}, t) dV = \int_{\Omega} \frac{\partial \rho}{\partial t}(\vec{x}, t) dV \quad (2.1)$$

Given a velocity field $\vec{u}(\vec{x}, t)$, let $(\rho\vec{u} \cdot \vec{n})$ be the mass flow rate across the boundary, then, the conservation of mass is :

$$\int_{\Omega} \frac{\partial \rho}{\partial t}(\vec{x}, t) dV = - \int_{\partial\Omega} (\rho\vec{u} \cdot \vec{n}) dA = \int_{\Omega} \vec{\nabla} \cdot (\rho\vec{u}) dV \quad (2.2)$$

If we combine the first and the third terms, we get:

$$\int_{\Omega} \left\{ \frac{\partial \rho}{\partial t} - \vec{\nabla} \cdot (\rho\vec{u}) \right\} dV = 0 \quad (2.3)$$

The equation could be understood as: the rate of change of mass in the volume Ω equals the net mass flow across the boundary for any arbitrary region $\Omega \subseteq D$.

2.1.2 Conservation of momentum

Applying Newton's second law to a small control volume in Cartesian coordinates, as shown in Figure 2.1, gives:

$\sum F_x$ on the CV = rate of outflow of x-wise momentum from the CV -
rate of inflow of x-wise momentum into the CV +
rate of accumulation of x-wise momentum inside the CV.

Figure 2.1: Control volume in Cartesian coordinates

where:

ρ'_x, ρ'_y : the density partial derivative with respect to x and y coordinates.

u, v : velocity components in x and y directions.

The mass of fluid flow through face BC could be written as:

$$\left[\left(\rho + \frac{\partial \rho}{\partial x} \frac{\Delta x}{2} \right) \left(u + \frac{\partial u}{\partial x} \frac{\Delta x}{2} \right) \right] \times \left[\left(u + \frac{\partial u}{\partial x} \frac{\Delta x}{2} \right) (\Delta y \times 1) \right] \quad (2.4)$$

Similarly, we could write fluid flow crossing face AB, AD, and CD. Substituting

these terms in the right hand of the above equation. Dividing through by the volume ($\Delta x \Delta y \times 1$) and neglecting terms which are second order or higher, we get the following on the right side of equation:

$$\rho \frac{\partial u}{\partial t} + \rho u \frac{\partial u}{\partial x} + \rho v \frac{\partial u}{\partial y} + u \left[\frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} + \frac{\partial \rho}{\partial t} \right] \quad (2.5)$$

From the continuity equation, the term $\left[\frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} + \frac{\partial \rho}{\partial t} \right]$ goes to zero. The right side of equation leaves us:

$$\rho \frac{\partial u}{\partial t} + \rho u \frac{\partial u}{\partial x} + \rho v \frac{\partial u}{\partial y} \quad (2.6)$$

By inspection, we could write similar equation for a three dimensional flow in x direction:

$$\rho \frac{\partial u}{\partial t} + \rho u \frac{\partial u}{\partial x} + \rho v \frac{\partial u}{\partial y} + \rho w \frac{\partial u}{\partial z} = \rho \frac{Du}{Dt} \quad (2.7)$$

Figure 2.2: Stresses acting on a differential volume

From solid mechanics, the shear stresses, Figure 2.2, have the following relations:

$$\tau_{xy} = \tau_{yx}; \quad \tau_{xz} = \tau_{zx}; \quad \tau_{yz} = \tau_{zy}$$

Therefore, the momentum equation in x direction could be written as:

$$\frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} = \rho \frac{Du}{Dt} \quad (2.8)$$

Considering a large number of unknowns, the stress-displacement relations for Newtonian fluid (in x-direction) are introduced:

$$\tau_{xy} = \mu \left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right) \quad (2.9)$$

$$\tau_{xz} = \mu \left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \right) \quad (2.10)$$

$$\tau_{xx} = -P + 2\mu \left(\frac{\partial u}{\partial x} \right) + \lambda \nabla \cdot \bar{V} \quad (2.11)$$

where λ is the second coefficient of viscosity. μ is the dynamic viscosity. P is the pressure. Substituting all of these expressions into the equation (2.8) gives the standard form of the Navier-Stokes equations:

$$\rho \frac{Du}{Dt} = \rho g_x - \frac{\partial P}{\partial x} + \frac{\partial}{\partial x} (2\mu \frac{\partial u}{\partial x} + \lambda \nabla \cdot \bar{V}) + \frac{\partial}{\partial y} (\mu (\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})) + \frac{\partial}{\partial z} (\mu (\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x})) \quad (2.12)$$

If the flow is incompressible, and if the viscosity, μ , is constant through the region of interest and the flow is steady, then the above equations can be written as:

$$u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} + w \frac{\partial u}{\partial z} = g_x - \frac{1}{\rho} \frac{\partial P}{\partial x} + \mu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) \quad (2.13)$$

where g_x is the body force in x direction.

Similar equations could be written for Navier-Stokes equation in y and z directions.

2.1.3 Conservation of energy

Let the kinetic energy of a differential volume at time t to be denoted as E_k , its internal energy is E_i , the net rate of heat flow into the particle by its surroundings is $\dot{Q}$, and the rate at which work is done on the particle by external forces is $\dot{W}$. Then, the conservation of energy can be written as:

$$\frac{D}{Dt}(E_i + E_k) = \dot{Q} - \dot{W} \quad (2.14)$$

Further insight into the energy equation can be obtained from thermodynamic principles. The application of the energy equation to droplets are detailed studied later.

2.2 Fluid flow equation

We consider the time-dependent nonlinear system:

$$\dot{\vec{u}} - \nu \Delta \vec{u} + (\nabla \vec{u}) \vec{u} + \nabla(\rho^{-1} p) = \rho^{-1} \vec{f} \quad \text{in } \Omega \quad (2.15)$$

$$\nabla(\vec{u}) = 0 \quad \text{in } \Omega \quad (2.16)$$

$$\vec{u} = \vec{g} \quad \text{on } \Gamma_D \quad (2.17)$$

$$\text{Either : } (-\rho^{-1}pI + \nu\nabla\vec{u})\vec{n} = \vec{h} \quad \text{on } \Gamma_N \quad (2.18)$$

$$\text{Or : } (-\rho^{-1}pI + 2\nu\Lambda\vec{u})\vec{n} = \vec{h} \quad \text{on } \Gamma_N \quad (2.19)$$

$$\vec{u} = \vec{u}_0 \quad \text{at } t = 0 \quad (2.20)$$

where $\vec{u}$ is the flow velocity, ρ is the density, $\nu = \mu/\rho$ is the kinetic viscosity (which may vary in time and space), Δ is the coordinate-wise Laplacian, Ω is a domain in R^3 or R^2 , Γ is its boundary, $\Gamma_D \cup \Gamma_N = \Gamma$, $\Gamma_D \cap \Gamma_N = \emptyset$, $\vec{g}$ is the Dirichlet data (a vector field defined on Γ_D , ordinarily 0, but there may be moving boundaries), $\vec{h}$ (or $\tilde{h}$) is the Neumann data (a vector field defined on Γ_N), I is the 3×3 identity matrix, $\vec{n}$ is the outward unit normal to Γ , $\nabla\vec{u}$ is the 3×3 matrix, $\{\partial u^i/\partial x^j\}$, and $\Lambda\vec{u}$ is $(\nabla\vec{u} + \nabla\vec{u}^T)/2$.

Since we are interested in thermal-transport problem, the heat equation is added:

$$\dot{T} - \alpha\Delta T + \vec{u} \cdot \nabla T = s \quad \text{in } \Omega \quad (2.21)$$

$$T = T_0 \quad \text{at } t = 0$$

$$T = \bar{T} \quad \text{at } \partial_1\Omega$$

$$\alpha\nabla T \cdot \vec{n} = z \quad \text{on } \partial_2\Omega$$

where T is temperature, α is thermal diffusivity, s is the source term, $\bar{T}$ and z are given functions on $\partial_1\Omega$ and $\partial_2\Omega$, $\dot{T}$ is the time derivative of temperature.

The two alternative forms, equations (2-18) and (2-19), of the Neumann boundary condition correspond to “pseudo-tractions” in the case of (2-18), and to actual

physical tractions, in (2-19).

There are some restrictions on the parameters of the problem:

1. The initial velocity must satisfy $\nabla(\vec{u}_0) = 0$ on Ω , and $\vec{n} \cdot \vec{u}_0 = \vec{n} \cdot \vec{g}$ on Γ_D .

Ordinarily, we would expect $\vec{u}_0 = \vec{g}$ on Γ_D , i.e., the initial velocity satisfies the same boundary condition as the solution. Note that $\vec{g}$ may be time-dependent.

2. In the case $\Gamma_D = \Gamma$, the Dirichlet data $\vec{g}$ must satisfy : $\int_{\Gamma} \vec{n} \cdot \vec{g} = 0$.

2.3 Free surface equation

In a filling analysis, there is special interest on the research of the physics of the boundary between liquid metal and the air. This boundary is often called free surface. If the fluid is in motion, the free surface is not in thermodynamic and mechanical equilibrium. However, we could study the transport of momentum across the surface and evolution of the surface when this local departure from equilibrium is not too large. This effect actually tends to make the velocity and momentum across the free surface uniform, which results in the transport of momentum and heat.

2.3.1 Surface tension

The fluid system can be shown schematically in the first picture of Figure 2.3. A small volume of this fluid system is denoted by Ω , and Γ is the boundary of the volume.

Both Ω and Γ are fixed in space. The free surface will be denoted by Π . Across the free surface separating the two fluids, as shown in the second picture, the velocity $\vec{u}$ is continuous, $\vec{u}_g = \vec{u}_l$, where $\vec{u}_g$ is gas velocity and $\vec{u}_l$ is the liquid velocity. The stresses act on both sides of the free-surface. Their relationship is established as follows in order to study the flux of momentum across the surface.

$$\vec{\sigma}_l \cdot \vec{n} - \vec{\sigma}_g \cdot \vec{n} = \tau \left(\frac{1}{R_1} + \frac{1}{R_2} \right) \quad (2.22)$$

Figure 2.3: The free surface and the surface tension

where τ is the surface tension coefficient, and R_1 and R_2 are principal radii of curvature. We can tell from the above equation that the jump of tractions across the surface is entirely balanced by the effect of surface tension. The surface tension effects are critical for narrow cross-sections in casting problems, while it can be safely ignored in larger sections.

2.3.2 Numerical simulation

Volume-Of-Fluid (*VOF*) approach is the most widely used tracking method for modeling free surface flow. It is relatively easy to implement, topologically robust. All *VOF* methods begin by computing the volume fraction of the fluid in each element in the computational domain based on the interface geometry. The volume of fraction, *vof*, is usually defined as the ratio of the volume of fluid in the cell to the total cell volume. Hence, $vof = 0$ when the cell is empty of fluid, $vof = 1$ when the cell is filled with fluid, and $0 < vof < 1$ for a mixed cell containing a liquid/air interface. So, instead of using the true geometry of the interface, the volume fraction is used to track the interface indirectly through a discrete continuity equation.

For incompressible flow, the conservation of mass is equivalent to conservation of volume, so the motion of an interface in such a flow could be modeled by solving the conservation of the *vof*. As a result, a conservative finite difference method could be used to update the volume fractions, and adjusting for the numerical errors, the volume / mass of each fluid in the computation is conserved.

There are two basic tasks to be solved by the interface advecting method. First, the interface needs to be reconstructed by using the given values of *vof* in elements. Second, the *vof* in each element needs to be updated using data from the velocity field and interface representation.

Youngs [4] advanced the *VOF* approach with a method commonly referred to as

PLIC (piecewise linear interface calculation). PLIC applies sloped piecewise lines to reconstruct the interface. The accuracy of the PLIC reconstruction depends entirely on the method used to compute the slope of the lines. One main advantage of PLIC method is that it doesn't align the lines in the coordinate directions. In other words, the PLIC approximation is multi-dimensional and is therefore unrelated to the advecting mesh direction. The PLIC method is very powerful at solving for the large scale features of an interface, but it has difficulty in tracking the finer scale feature. This means that the PLIC method has a tendency to smooth high curvature regions. The template for our PLIC implementation is based on Youngs' method.

2.4 Droplet equations

In this section, the governing equations for a single particle or droplet will be presented. These equations are fundamental to the development of numerical models for dispersed-phase flows. These derivations result from the application of Reynolds Transport Theorem which is briefly presented next.

2.4.1 Reynolds transport theorem

Reynolds Transport Theorem provides the link between the Lagrangian and Eulerian approaches to study the dynamic and thermal state of a fluid. If we assume B is an

extensive property of a fluid and β is the corresponding intensive property, similarly to mass net flux equation, the net efflux of property B from the volume is:

$$\dot{B} = \int_S \rho \beta \vec{w}_i \vec{n}_i dS \quad (2.23)$$

An arbitrary control volume in space through which fluid can pass is shown in the Figure 2.4. It can move through space, rotate or change size.

Figure 2.4: System moving through a control volume at two successive time

Now consider a mass element (referred as “system”) which moves through the control volume. At time t , the boundaries of the system are the control surface. At this time, the mass of the system is the mass in the control volume. At time $t + \delta t$, part of the system moves out of the control volume. The mass of the system which has left the

volume is δm_{out} , while the mass which has entered the volume is δm_{in} .

The amount of property B associated with the mass (system) has left is δB_{out} , and the corresponding amount of B with mass that has entered is δB_{in} . By definition, the rate of change of property B of the system is given by:

$$\frac{dB_{sys}}{dt} = \lim_{\delta t \rightarrow 0} \frac{B_{sys}(t + \delta t) - B_{sys}(t)}{\delta t} \quad (2.24)$$

The amount of B of the system at time t corresponds to the amount of B in the control volume at time t :

$$B_{sys}(t) = B_{cv}(t) \quad (2.25)$$

At time $t + \delta t$:

$$B_{sys}(t + \delta t) = \delta B_{out} + B_{cv}(t + \delta t) - \delta B_{in} \quad (2.26)$$

Substitute the expressions for B_{sys} at t and $t + \delta t$ into equation (2-24), rearrange:

$$\frac{dB_{sys}}{dt} = \lim_{\delta t \rightarrow 0} \left[\frac{B_{cv}(t + \delta t) - B_{cv}(t)}{\delta t} + \frac{\delta B_{out}}{\delta t} - \frac{\delta B_{in}}{\delta t} \right] \quad (2.27)$$

The first two terms are simply the rate of change of the property B in the control volume:

$$\lim_{\delta t \rightarrow 0} \left[\frac{B_{cv}(t + \delta t) - B_{cv}(t)}{\delta t} \right] = \frac{dB_{cv}}{dt} \quad (2.28)$$

The amount of B in the control volume is obtained by integrating the product of the corresponding intensive property β and fluid density over the control volume.

Thus,

$$\frac{dB_{cv}}{dt} = \frac{d}{dt} \int_{cv} \rho \beta dV \quad (2.29)$$

The other two terms represent the net efflux of property across the control surface:

$$\lim_{\delta t \rightarrow 0} \left[\frac{\delta B_{out}}{\delta t} - \frac{\delta B_{in}}{\delta t} \right] = \int_{cs} \rho \beta \vec{w}_i \vec{n}_i dS \quad (2.30)$$

Finally, Reynolds transport theorem becomes:

$$\frac{dB_{sys}}{dt} = \frac{d}{dt} \int_{cv} \rho \beta dV + \int_{cs} \rho \beta \vec{w}_i \vec{n}_i dS \quad (2.31)$$

where $\vec{w}_i$ is the velocity of fluid with respect to control surface. $\vec{n}_i$ is the normal vector of the control surface.

2.4.2 Single particle equation

Reynolds transport theorem will be used to derive the fundamental equations for dispersed phase flows. We start from the equations describing the conservation of mass, momentum and energy for an individual droplet or particle. They might be regarded as “Lagrangian” because they apply to moving mass elements and do not describe the change of conditions at a spatial point. In Figure 2.5, $\vec{v}_i$ is defined as the particle velocity with respect to an inertial reference frame. r is the radius of the particle. The particle is assumed to be spherical and assumed not to rotate. The control surface lies within the boundary layer of the particle. The transition from particle surface properties to local fluid properties takes place across the boundary

layer. The velocity $\vec{w}_i$ is the velocity through the control surface with respect to the control surface.

Figure 2.5: Moving control surface enclosing particle

Mass conservation

Applying Reynolds transport theorem to mass conservation, we get:

$$\frac{d}{dt} \int_{cv} \rho_d dV + \int_{cs} \rho_s \vec{w}_i \cdot \vec{n}_i dS = 0 \quad (2.32)$$

where ρ_d is the material density of the particle and ρ_s is the density of the fluid at the control surface. The integral of the density over the volume for the particle is simply the instantaneous mass of the particle, m . Also if we assume $\vec{w}_i$ and ρ are uniform

over the control surface, the continuity equation can be written:

$$\frac{dm}{dt} = -\rho_s w S_d \quad (2.33)$$

where w is the magnitude of the efflux velocity vector, $\vec{w}_i \cdot \vec{n}_i$. If the particle is evaporating, w is positive and the particle mass decreases with time. S_d is the surface area of the particle.

Linear momentum conservation

From Newton's Second Law:

$$\vec{F} = \frac{d(M\vec{U}_i)}{dt} \quad (2.34)$$

In this case, the intensive variable is $\vec{U}_i$. The application of Reynolds transport theorem for rate of change of momentum yields:

$$\vec{F}_i = \frac{d}{dt} \int_{cv} \rho_d \vec{U}_i dV + \int_{cs} \rho_s \vec{U}_{i,s} \vec{n}_j \vec{w}_j dS \quad (2.35)$$

where $\vec{U}_{i,s}$ is the velocity of fluid at the control surface with respect to the inertial reference frame. This is a vector equation for the direction i . Considering the rotation of a droplet (while ignoring any internal circulation), we have the velocity inside the particle as:

$$\vec{U}_i = \vec{v}_i + \epsilon_{ijk} \vec{w}_j \xi_k \quad (2.36)$$

where ϵ_{ijk} is the alternating tensor, $\vec{v}_i$ is the velocity of the particle's center of mass with respect to an inertial reference frame, $\vec{w}_j$ is particle's rotation vector and ξ_k is

the distance from the center of mass. Considering $\vec{v}_i$ and $\vec{\omega}_i$ are constant over the interior of the particle. The integral for the momentum in the control volume can be written as:

$$\frac{d}{dt} \int_{cv} \rho_d \vec{U}_i dV = \frac{d}{dt}(\vec{v}_i m) + \frac{d}{dt} \epsilon_{ijk} \vec{w}_j \int_{cv} \xi_k \rho_d dV \quad (2.37)$$

By definition of the center of mass, the second term on the right hand side of the above equation is zero, so the equation can be reduced to:

$$\frac{d}{dt} \int_{cv} \rho_d \vec{U}_i dV = \frac{d}{dt}(\vec{v}_i m) = \vec{v}_i \frac{dm}{dt} + m \frac{d\vec{v}_i}{dt} \quad (2.38)$$

If the particle is evaporating or condensing, then the velocity of the surface with respect to the center of the particle is $\dot{r}\vec{n}_i + \epsilon_{ijk}\vec{\omega}_j r_k$ where $\dot{r}$ is the regression rate of the surface. Thus the velocity of the fluid crossing the control surface with respect to an inertial reference frame is:

$$\vec{U}_{i,s} = \vec{v}_i + \epsilon_{ijk}\vec{\omega}_j r_k + (\dot{r} + \omega)\vec{n}_i = \vec{v}_i + \epsilon_{ijk}\vec{\omega}_j r_k + w'\vec{n}_i \quad (2.39)$$

where, for convenience, the sum of the regression rate and efflux velocity, $(\dot{r} + \omega)\vec{n}_i$ has been replaced by $w'\vec{n}_i$. The net efflux of momentum from the control surface is given by:

$$\int_{cs} \rho_s \vec{U}_{i,s} \vec{w}_j \vec{n}_j dS = \int_{cs} \rho_s (\vec{v}_i + \epsilon_{ijk}\vec{\omega}_j r_k + w'\vec{n}_i) \vec{w}_j \vec{n}_j dS \quad (2.40)$$

In that $\vec{v}_i$ is not a function of the surface position, using continuity equation, one can write:

$$\vec{v}_i \int_{cs} \rho_s \vec{w}_j \vec{n}_j dS = -\vec{v}_i \frac{dm}{dt} \quad (2.41)$$

Finally the momentum equation for a particle becomes:

$$\vec{F}_i = m \frac{d\vec{v}_i}{dt} + \int_{cs} \rho_s w' \vec{n}_i \vec{w}_j \vec{n}_j dS + \epsilon_{ijk} \vec{w}_j \int_{cs} \rho_s r_k \vec{w}_j \vec{n}_j dS \quad (2.42)$$

Energy Conservation

The first law for thermodynamics is stated as:

$$\frac{dE}{dt} = \dot{Q} - \dot{W} \quad (2.43)$$

where $\dot{Q}$ and $\dot{W}$ are the change rates at the instant the system occupies the control volume. E is total energy, which could be expressed as:

$$E = M \cdot e + S \cdot \delta \quad (2.44)$$

where e is the sum of specific internal energy and kinetic energy. δ is the surface tension. Applying Reynolds Transport theorem to the energy equation gives:

$$\dot{Q} - \dot{W} = \frac{d}{dt} \left[\int_{cv} \rho_d e dV + \int_{cs} \rho_s e_s \vec{w}_j \vec{n}_j dS \right] + \frac{d}{dt} (S \cdot \delta) \quad (2.45)$$

Assuming the particle is neither rotating nor has internal motion and the velocity of matter inside the droplet is $\vec{v}_i$. The rate of change of the energy within the control volume (the first and the third term on the right of the equation) becomes:

$$\begin{aligned} \frac{d}{dt} \int_{cv} \rho_d e dV + \frac{d}{dt} (S \delta) &= \frac{d}{dt} \int_{cv} \rho_d \left(\bar{e} + \frac{\vec{v}_i \vec{v}_i}{2} \right) dV + \delta \frac{dS}{dt} \\ &= \frac{d}{dt} \left(m \frac{\vec{v}_i \vec{v}_i}{2} \right) + \frac{d}{dt} (m \bar{e}) + \frac{2\delta}{r} \dot{r} S \end{aligned} \quad (2.46)$$

where, $\bar{i}$ is the average specific internal energy of the particle and the surface tension is assumed constant with time. Next, we will evaluate the second term in equation (2.45), i.e., energy flux term:

$$e_s = i_s + \frac{\vec{U}_{i,s} \vec{U}_{i,s}}{2} \quad (2.47)$$

where i_s is the specific internal energy at the surface, $\vec{U}_{i,s}$ is the velocity with respect to an inertial reference frame. This velocity can be expressed as:

$$\vec{U}_{i,s} = \vec{v}_i + \vec{w}_i + \dot{r} \vec{n}_i = \vec{v}_i + \vec{w}'_i \quad \text{here} \quad \vec{w}'_i = (w + \dot{r}) \vec{n}_i \quad (2.48)$$

Substitute $\vec{U}_{i,s}$ and e_s into the energy flux term and rearrange, we notice the integral of the mass flux over the surface is equal to $-(dm/dt)$:

$$\int_{cs} \rho_s e_s \vec{w}_j \vec{n}_j dS = -\left(\frac{\vec{v}^2}{2} + \frac{w'^2}{2}\right) \frac{dm}{dt} + \vec{v}_i \int_{cs} \rho_s w' w \vec{n}_i dS + \int_{cs} i_s \rho_s \vec{w}_j \vec{n}_j dS \quad (2.49)$$

Adding equations (2-46) and (2-49) to get the rate of change of energy of the system at the instant it is in the control volume:

$$\frac{dE}{dt} = m \bar{v}_i \frac{d\vec{v}_i}{dt} - \frac{(w')^2}{2} \left(\frac{dm}{dt}\right) + \frac{d}{dt}(m \bar{i}) + \vec{v}_i \int_{cs} \rho_s w' w \vec{n}_i dS + \int_{cs} \rho_s i_s \vec{w}_j \vec{n}_j dS + \frac{2\delta}{r} \dot{r} S \quad (2.50)$$

Heat transfer to particle

There are two forms of heat transfer between particles: radiation and conduction. If we assume the particle is a grey body, the net radiative heat transfer to the particle is:

$$\dot{Q}_r = 4\pi r^2 (\alpha_1 J - \epsilon \sigma_1 T_d^4) \quad (2.51)$$

where α_1 is the absorptivity, ϵ is the emissivity, J is the radiosity and σ_1 is the Stephan-Boltzmann constant.

The heat transfer through net heat conduction could be obtained by integrating the heat flux vector, $\vec{q}_{i,c}$, over the surface of the particle as follows:

$$\dot{Q}_c = - \int_{cs} \vec{q}_{i,c} \vec{n}_i dS \quad (2.52)$$

When $\dot{Q}_c$ is positive, the heat is transferred into the particle.

Work rate of particles on surroundings

Work done by the system on the surrounding arises from both body force and surface force. Only gravity is considered in body force here. The work rate due to gravity is as follows:

$$\dot{W}_b = -\vec{F}_{i,b} \vec{v}_i \quad (2.53)$$

The work rate due to pressure on a small area (dS) is the dot product of the pressure force acting on an element area and the velocity of the matter passing through the element. Integrating over the total control surface gives us the work rate over the control surface, we get the work done by the system on surrounding:

$$\dot{W}_p = \int_{cs} (\vec{v}_i + \vec{w}'_i) p_s \vec{n}_i dS \quad (2.54)$$

Similarly, the expression for the work done by the system due to shear force is:

$$\dot{W}_s = - \int_{cs} (\vec{v}_i + \vec{w}'_i) \tau_{ji} \vec{n}_j dS \quad (2.55)$$

Add the work rate due to pressure and shear force and notice the velocity $\vec{v}_i$ is constant, we get:

$$\int_{cs} (\vec{v}_i + \vec{w}'_i)(p_s \vec{n}_i - \tau_{ji} \vec{n}_j) dS = \vec{v}_i \int_{cs} (p_s \vec{n}_i - \tau_{ji} \vec{n}_j) dS + \int_{cs} p_s \vec{w}'_i \vec{n}_i dS - \int_{cs} \vec{w}'_i \tau_{ji} \vec{n}_j dS \quad (2.56)$$

The first integral on the right is the surface force acting on the droplet, $\vec{F}_s$.

The second integral could be rearranged as:

$$\int_{cs} p_s \vec{w}'_i \vec{n}_i dS = \int_{cs} p_s (\vec{w}_i + \dot{r} \vec{n}_i) \vec{n}_i dS = \int_{cs} \rho_s \frac{p_s}{\rho_s} \vec{w}_i \vec{n}_i dS + \int_{cs} \dot{r} p_s dS \quad (2.57)$$

The third integral is zero, because $\tau_{ji} \vec{n}_j$ lies in the plane tangential to the particle surface, while the vector $\vec{w}'_i$ is normal to the surface. Combining all related terms into energy equation and making proper rearrangement:

$$\begin{aligned} \frac{d}{dt}(m\bar{i}) + \vec{v}_i(m \frac{d\vec{v}_i}{dt} + \int_{cs} \rho_s w' w \vec{n}_i dS - \vec{F}_{i,b} - \vec{F}_{i,s}) - \frac{(w')^2}{2} \frac{dm}{dt} + \frac{2\delta}{r} \dot{r} S \\ + \int_{cs} (i_s + \frac{p_s}{\rho_s}) \rho_s \vec{w}_i \vec{n}_i dS = \dot{Q}_r + \dot{Q}_c - \int_{cs} \dot{r} p_s dS \end{aligned} \quad (2.58)$$

Refer to equation (2-42), assuming the particle does not rotate, the factor multiplying $\vec{v}_i$ in the second term on the left of the equation is zero. Also assuming the regression rate be uniform over the surface and the combination $\bar{p}_s + 2\sigma/r$ is the pressure inside the droplet, p_d . So the equation reduces to:

$$m \frac{d\bar{i}}{dt} + \bar{i} \frac{dm}{dt} - \frac{w'^2}{2} \frac{dm}{dt} + \int_{cs} h_s \rho_s \vec{w}_i \vec{n}_i dS = \dot{Q}_r + \dot{Q}_c - \dot{r} p_d \frac{p_d}{\rho_d} S \quad (2.59)$$

where $h_s = i_s + \frac{p_s}{\rho_s}$ is the enthalpy of the fluid crossing the control surface.

The factor $\rho_d \dot{r} S$ is the rate of change of droplet mass dm/dt . Also notice the kinetic energy of the mass efflux velocity is very small compared with the enthalpy difference between phases. Simplifying the equation gives the conventional form of the particle energy equation:

$$m \frac{d\bar{i}}{dt} = \dot{Q}_r + \dot{Q}_c + \frac{dm}{dt}(\bar{h}_s - \bar{h}_d) \quad (2.60)$$

where $\bar{h}_d = \bar{i} + \frac{p_d}{\rho_d}$.

2.4.3 Volume averaged droplet equation

In practice, we have to deal with a mixture of droplets. It is impractical to solve for the fluid properties at every point in the mixture. We have to consider the average properties in the flow and ascribe them to some point in the enclosed volume. Below, a formal approach is introduced to express the conservation laws for the continuous phase in terms of average properties.

Volume averaging

Consider a spherical volume (CV) that encloses a mixture of fluid and particles as shown in Figure 2.6. The average volume V is the summation of volume of continuous phase (V_c) and volume of dispersed phase (V_d). The average properties are ascribed to the center of the sphere. If we use B to represent some property of the continuous phase, the following two types of averages are considered.

The phase average $\langle B \rangle$ is the average over the volume of the continuous phase V_c :

$$\langle B \rangle = \frac{1}{V_c} \int_{V_c} B dV \quad (2.61)$$

While the local volume average $\overline{B}$ is defined over the average volume $V = V_c + V_p$:

$$\overline{B} = \frac{1}{V} \int_{V_c} B dV \quad (2.62)$$

Volume average of gradient operation and time derivative

Gradient and time derivative terms are always involved in the conservation equations for fluid flow. We need to form a relationship for local volume average of gradient and time derivative. Local volume average of gradient is defined as:

$$\overline{\frac{\partial B}{\partial x_i}} = \frac{1}{V} \int_{V_c} \frac{\partial B}{\partial x_i} dV \quad (2.63)$$

Local volume average of time derivative is defined as:

$$\overline{\frac{\partial B}{\partial t}} = \frac{\partial \overline{B}}{\partial t} \quad (2.64)$$

In Figure 2.6, we define:

Total surface area of the complete particles is S_e .

The surface area surrounding the particles inside the CV is S_d .

The interior surface area of the severed particles is S_b .

The control surface area of the severed particles is S_s .

The control surface area of the continuous phase is S_c .

The complete surface area of the control volume is S .

So, we have $S = S_c + S_s$ and $S_d = S_b + S_e$ hold.

Figure 2.6: Definition of surfaces

The volume average of the gradient is the gradient of the volume average plus a term which involves the integral of B over the interface between the dispersed and continuous phases.

$$\overline{\frac{\partial B}{\partial x_i}} = \frac{\partial \overline{B}}{\partial x_i} + \frac{1}{V} \int_{S_d} B n_i dS \quad (2.65)$$

The volume average of the time derivative is given by:

$$\overline{\frac{\partial B}{\partial t}} = \frac{\partial \overline{B}}{\partial t} + \frac{1}{V} \int_{S_d} B \dot{r} dS \quad (2.66)$$

These equations will be used in the expressing of the volume-averaged forms of conservation laws, whose derivations are omitted here.

2.5 Coupled differential equations for casting analysis

In casting analysis, the physics of melting and solidification, evolution of the free-surface, flow field and the thermal field present an intricately coupled, complex and nonlinear system of differential equations. The coupled equations may be classified at two different levels: 1) In casting analysis, most of the problems involve multiphase, multicomponent systems. These phases and components interact and they are strongly coupled. 2) Coupling at the level of state variables, i.e., $\vec{u}, p, t, \rho$, and systems of equations. The relations of the state variables at this level of coupling are usually decomposed with regard to some simplifying assumptions and grouped under physically meaningful expressions such as Navier-Stokes equation, energy equation, free-surface equation, etc. Careful examination of the physics may identify the weaker couplings between these equations and may allow the equations to be decoupled.

The nature of the physical problem and the mathematical formulation of the problem determines the coupled equations governing the phenomena. For example, if heat flow is studied in a domain with steady motion, only the energy equation

governs the thermal flow; no correspondence between the energy equation and the Navier-Stokes equation of fluid mechanics is required. In this sense, the Navier-Stokes and the energy equations are decoupled and the constant velocity field is used in the energy equation. However, in most realistic problems, the system of equations is coupled. Often, the coupling is critical.

To analyze a wide range of problems under varying conditions, such as those encountered by the casting designers, methods and tools must be presented to fit their specific analysis requirements. The numerical examples demonstrate some coupled systems could be found in [6].

Chapter 3

Droplet Filling Solver

An ideal filling solver will conserve mass, momentum and energy exactly and satisfy constitutive equations and boundary conditions. Particle methods such as Sulsky's [13] can conserve mass, momentum and energy exactly.

3.1 A review of a particle method

Sulsky's Euler-Lagrange method uses Lagrangian particles in which the mass of each particle usually remains constant. These particles are advected in a continuous displacement, velocity, acceleration fields that are defined on a C^0 mesh, e.g., using 8-node bricks. Each particle carries its full state of mass, energy, momentum, velocity, vorticity, etc. Material particles are purely Lagrangian and the mesh can be viewed as Eulerian in the sense that it maps data from a set of material points, the

particles, to a spatial grid and not Eulerian in the sense that in the next time step, the mesh nodes move with the velocity field, i.e., the mesh now behaves as an updated Lagrangian mesh. Sulsky's Euler-Lagrange method has the remarkable property of conserving mass, energy and momentum exactly if the consistent mass matrix is used. The primary restriction on the remeshing algorithm is that each element in the mesh must have enough material points, almost exactly in the same sense that a standard FEM element must have enough Gauss points, to properly constrain the null space. Except for this point, the mesh can be quite arbitrary.

If the flow is inertia dominated, then a useful predictor for Lagrangian advection is to advect particles individually using an appropriate equation of motion. In this way, most particle interaction terms are ignored and governing equations are simplified.

3.2 Single droplet test

The basic governing equation is tested on the movement of a single droplet in this section. The constant field velocity is assumed to be the positive x-direction. Positive y direction is defined as the direction of projected component of particle velocity on axis perpendicular to x-axis. The particle is assumed to be a sphere, moving in the direction with an inlet angle with the velocity field.

Mass of the particle:

$$M_p = \rho_p \times \left(\pi \frac{d_p^3}{6}\right) \quad (3.1)$$

3.2.1 The movement equations

When the particle is taken as reference system, the drag force on the particle in x direction could be computed from:

$$D_x = C_D \cdot \frac{PV_{rx}^2}{2RT} \cdot \frac{\pi d_p^2}{4} \quad (3.2)$$

where:

C_D : the drag coefficient.

P: the average gas pressure.

R: air gas constant, taken as $287 \text{ N} \cdot \text{m}/\text{kg} \cdot \text{K}$.

T: the average gas temperature.

V_{rx} : the relative velocity of the field with respect to the particle.

d_p : the particle diameter.

The driving force (positive X-direction) is equal to:

$$F_x = M_p \cdot a = \rho_p \cdot \frac{\pi d_p^3}{6} \cdot \frac{dV_{rx}}{dt} \quad (3.3)$$

Because the gravity force on particle is small, $M_p \cdot g$ is neglected. Equating driving force with the drag force gives:

1. When the relative velocity of the particle to the field $d\vec{v}_{rx}$ is positive, its expression is:

$$dV_{rx} = \frac{3 \cdot C_D \cdot P \cdot V_{rx}^2}{4 \cdot R \cdot T \cdot \rho_p \cdot d_p} \cdot dt \quad (3.4)$$

2. When the relative velocity of the particle to the field $d\vec{v}_{rx}$ is negative, its expression is:

$$dV_{rx} = -\frac{3 \cdot C_D \cdot P \cdot V_{rx}^2}{4 \cdot R \cdot T \cdot \rho_p \cdot d_p} \cdot dt \quad (3.5)$$

Superimpose V_{rx} with the particle velocity component in x direction to get the particle absolute velocity, V_x . Because $\frac{dx}{dt} = V_x$, then we get the distance of particle movement $dx = V_x \cdot dt$.

Similarly, if the relative velocity of field to particle in Y-direction is V_{ry} , then:

Drag force:

$$D_y = C_D \cdot \frac{P \cdot V_{ry}^2}{2 \cdot R \cdot T} \cdot \frac{\pi d_p^2}{4} \quad (3.6)$$

Driving force (positive Y-direction):

$$F_y = M_p \cdot a = \rho_p \cdot \frac{\pi d_p^3}{6} \cdot \frac{dV_x}{dt} \quad (3.7)$$

Equate driving force with the drag force. Considering that dV_{ry} is always in negative Y-direction:

$$dV_{ry} = -\frac{3 \cdot C_D \cdot P \cdot V_{ry}^2}{4 \cdot R \cdot T \cdot d_p} \cdot dt \quad (3.8)$$

Because $\frac{dy}{dt} = V_y$, the distance this particle moves in y direction $dy = V_y \cdot dt$.

3.2.2 Choosing parameters

The air gas constant, R , is $287 \text{ N}\cdot\text{m}/\text{Kg}\cdot\text{K}$. The average gas temperature, T , is taken as 700K . The density of the liquid Aluminum, ρ_p , is $2390.0 \text{ kg}/\text{m}^3$. For test purpose, the value of average gas pressure, P , dynamic viscosity, μ , and particle diameter, d_p , is magnified and taken as 20 MPa , $3.55 \text{ e}^{-3} \text{ N}\cdot\text{s}/\text{m}^2$, and 0.02 m respectively.

Special attention should be given to choose a proper value of C_D . In the droplet trajectory calculation, the flow field is known as a discrete (mesh) form. The particle equations of motion are integrated numerically using appropriately small steps. The predominant force acting on a particle is the drag force, C_D , whose value depends mainly on the particle Reynolds number, Re_p . The dependence on Re_p asks for accurate $C_D - Re_p$ relationship [9]:

$$C_D = 0.4 + (24/Re_p) + \frac{6}{1 + \sqrt{Re_p}} \quad (3.9)$$

where Re is determined by:

$$Re = \rho_p \times d_p \times |\vec{V}_f - \vec{V}_p|/\mu \quad (3.10)$$

where:

V_f : velocity of field;

V_p : velocity of particle;

3.2.3 Test results

A droplet is advected by ODE solver for a certain time interval, 0.1 s . After each 0.01 s , the droplet velocity is recorded and compared with results from theoretical calculation. Different conditions are applied on the particle in the following numerical tests. The inlet angle of the particle moving with respect to the air field is represented by α .

- **Test I – droplet moving with zero relative velocity**

Conditions:

$$V_f = [5, 0, 0] \text{ m/s}$$

$$V_p = [5, 0, 0] \text{ m/s}$$

$$\alpha = 0^\circ$$

Time	Velocity from ODE solver	from theoretical calculation
0.010	5.0	5.0
0.020	5.0	5.0
0.030	5.0	5.0
0.040	5.0	5.0
0.050	5.0	5.0
0.060	5.0	5.0
0.070	5.0	5.0
0.080	5.0	5.0
0.090	5.0	5.0
0.100	5.0	5.0

Table 3.1: Test I – droplet movement with zero relative velocity

- **Test II – droplet velocity being decelerated by the air**

Conditions:

$$V_f = [5, 0, 0] \text{ m/s}$$

$$V_p = [10, 0, 0] \text{ m/s}$$

$$\alpha = 0^\circ$$

Time	Velocity from ODE solver	from theoretical calculation
0.010	9.8346	9.8346
0.020	9.6799	9.6799
0.030	9.5348	9.5348
0.040	9.3984	9.3984
0.050	9.2700	9.2700
0.060	9.1489	9.1489
0.070	9.0344	9.0344
0.080	8.9262	8.9261
0.090	8.8235	8.8234
0.100	8.7261	8.7260

Table 3.2: Test II – droplet velocity being decelerated by the air

- **Test III – droplet velocity being accelerated by the air**

Conditions:

$$V_f = [10, 0, 0] \text{ m/s}$$

$$V_p = [5, 0, 0] \text{ m/s}$$

$$\alpha = 0^\circ$$

Time	Velocity from solver	from theoretical calculation
0.010	5.1653	5.1653
0.020	5.3201	5.3201
0.030	5.4652	5.4652
0.040	5.6016	5.6016
0.050	5.7299	5.7299
0.060	5.8511	5.8511
0.070	5.9655	5.9655
0.080	6.0739	6.0739
0.090	6.1766	6.1765
0.100	6.2741	6.2740

Table 3.3: Test III – droplet velocity being accelerated by the air

- **Test IV – droplet moving with 45° inlet angle**

Conditions:

$$V_f = [0, 0, 0] \text{ m/s}$$

$$V_p = [5, 5, 0] \text{ m/s}$$

$$\alpha = 45^\circ$$

Time	Velocity U from ODE solver	from theoretical calculation
0.010	4.8361	4.8361
0.020	4.6827	4.6827
0.030	4.5387	4.5387
0.040	4.4034	4.4034
0.050	4.2759	4.2759
0.060	4.1556	4.1556
0.070	4.0418	4.0418
0.080	3.9342	3.9342
0.090	3.8321	3.8321
0.100	3.7353	3.7352

Table 3.4: Test IV – droplet moving with 45° inlet angle

Chapter 4

Portrait of Solvers

4.1 Droplet filling solver

There are two types of droplets, *inletDroplet* and *vofDroplet*. An *inletDroplet* is generated based on our specifications and always travels from an initial position at the inlet boundary of the filling domain. A *vofDroplet* is generated by the solver based on the *vof* field. It might start its travel from any initial position in the filling domain.

4.1.1 Determine inletDroplet number and size

If we grow one layer from *shotSleeve* into *runnerGate*, the number of *inletDroplet* could be taken as the number of elements in this layer. The diameter of each droplet

can be determined based on the geometric size of each element, $d = \sqrt{4a^2/\pi}$. However, we could also make a bigger droplet if a fine mesh is used in order to decrease the number of droplets.

4.1.2 Determine startTime & endTime

There are different clocks for different solvers to record the advance of local time, but there is only one global time clock. For droplet solver, the start time of a specific step is the current global time, while the end time is obtained as:

$$EndTime = StartTime + ODETimeInterval .$$

here, the *ODETimeInterval* is computed from the amount of metal liquid we expect to fill in each time step. We set *nbLayersToAdvect* at the beginning, then:

$$ODETimeInterval = \frac{nbLayersToAdvect \times elementDimension}{dropletInletVelocity} .$$

4.1.3 Making the bounce element layer

Find all the boundary elements of the filling domain, and store them in the *ODEDict* under the field name: *BounceElementIds*. These elements enclose the filling domain. Droplets could bounce back and forth between walls while they are traveling in the domain, but they can never go beyond this wall and get out of the filling domain.

4.1.4 Droplet state parameters

A field named *parameters* should be created in the *ODEDict*. The parameters include: *density*, *dynamicViscosity*, *temperature*, *pressure*, and *bounceLossFactor*.

These scalar parameters describe the filling conditions of metal liquid. The *bounceLossFactor* varies from 0.0 to 1.0. If a pure elastic bounce happens between the die wall and a droplet, this factor should be 1.0. In addition, a field named *dropletDiameter* should be created and populated in the *ODEDict* before solving *inletDroplet* ODESolver. The dimension of this field is the same as the number of *inletDroplets*.

4.1.5 Droplet initial parameters

The initial parameters for droplets are specified in *startValues* array. The state of each droplet is described by three initial coordinates, three initial velocity components and one initial total enthalpy. The initial coordinates can be taken to be the centroid position of an inlet element. While the initial velocity and initial enthalpy are read from the design file.

4.1.6 Advect inlet droplets

There are five options for the current ODEsolver, *FlowLine*, *DropletVof*, *DropletInlet*, *DropletMixed*, *FluxLine*. First *DropletInlet* option is instantiated in the ODEsolver to advect inlet droplets. In any element that an inlet droplet passes through, a

vofDroplet will be instantiated if some lumped liquid point falls in it.

4.1.7 Output from inlet droplet solver

The result from the inlet droplet solver includes:

- *incrMass*: One real number for one element. It reports the amount of liquid increased in an element for this *ODETimeInterval*.
- *incrMomentum*: A vector with three real numbers for one element. It gives the change of momentum in an element in three directions.
- *coordinates*: Gives the position where droplets halt. These real numbers are in an array.
- *dropletCoordinates*: Three real numbers for one element. Gives the weighted average coordinates of the liquid droplets in one element.
- *dropletVelocity*: Gives the droplet velocity when it halts. The velocity components are arranged in the form of an array.

4.1.8 Map results

In order to analyze and visualize the flow pattern of metal liquid, the droplet data must be mapped to *vof* and velocity. Dividing the *incrMass* in an element by the

element volume, the element *vof* is obtained. Similarly, dividing the *incrMomentum* of an element by the element total mass gives the velocity vector for liquid in the element. After obtaining *vof* and velocity, DFP (Divergence Free Projection) solver could be solved to make a div-free velocity field and ERK solver to compute velocity and pressure and to conserve the momentum.

4.1.9 Distribute extra vof

The current droplet solver does not enforce the incompressibility condition of metal liquid after droplet moving and data mapping, i.e., the *vof* could be larger than one. This happens when droplets pile up at corners. In this case, we have to distribute the extra *vof* (*vof* larger than one) in an element. There are several ways to achieve this objective. Two schemes considered in the solver are as follows:

1. Loop over elements, find the set of elements with $vof \geq 1.0$ and the set of elements with $vof \geq 0.0$. Then grow layer by layer from the set of elements with $vof \geq 1.0$ into the set of elements with $vof \geq 0.0$. After growing each layer, turn layer elements from zero *vof* to $vof = 1.0$ and subtract the corresponding amount of liquid from the total extra *vof*. Keeping growing layers until all extra *vof* is gone. Distributing extra *vof* in this way is easy to code and takes less computer time, but often this makes the flow pattern a little unrealistic, because extra *vof* is not properly distributed in the metal liquid flow direction.

2. Distribute extra *vof* downstream in the metal liquid flow direction. First, make *extraVofDroplet* from *vof*-larger-than-one elements. Advect these *extraVofDroplet* as *vofDroplet* using ODE solver, find the elements that these *extraVofDroplet* pass by, name them *passedElSet*. Try to distribute extra *vof* to empty space in this *passedElSet*, keep making new *extraVofDroplets* and advecting in this way until all extra *vof* has been scattered. Distributing extra *vof* in this way produces more reasonable results, but *extraVofDroplet* usually does not have proper velocity direction to travel around the corners in a complex geometry die. As a result, DFP has to be solved many times. It may become inefficient to distribute extra *vof* in this way and, sometime, even not possible to scatter all extra *vof*.

Later, if we include a Newmark method in the solver, the incompressibility could be satisfied automatically.

4.1.10 Make new droplets from vof

After extra *vof* has been distributed completely and incompressibility condition is satisfied, we could copy data in the current *vof* field and velocity field to result file for postprocessing later. Then make new *vofDroplets* from the current filled metal liquid. A new droplet will be generated from each nonzero *vof* element, and the droplet size will depend on the amount liquid contained in this element. The

new *vofDroplet* initial position is the weighted coordinate average of lumped liquid points whose centroids are in this element. The initial velocity is read from the droplet velocity field. Either DFP or ERK is solved on this velocity field each step to help droplets travel around the corners.

4.1.11 Advect vofDroplet

The option *DropletVof* should be instantiated in the ODE solver. As mentioned earlier, the behavior of a *vofDroplet* is different from an *inletDroplet*. A *vofDroplet* only records where it halts. Then the liquid in this *vofDroplet* is mapped to the element in which it halts.

From the above procedure, we can see each step of droplet filling is composed of two parts, advecting *inletDroplets* and advecting *vofDroplets*, which are done alternately.

4.1.12 Different versions of droplet filling solvers

Currently, we have three versions of droplet filling solvers. They are:

1. droplet solver
2. droplet + DFP solver
3. droplet + DFP + ERK solver

We use droplet ODE solver to advect droplets (metal liquid). DFP takes velocity as input and computes a divergence free velocity field, while ERK solves 3D transient Navier-Stokes equations for incompressible viscous flow to conserve the momentum. Considering ERK takes more computer time than DFP, dropletDFP solver is usually preferred. However dropletDFPERK solver is expected to provide a better velocity field. The reason that we solve DFP in droplet solver is to reduce the tendency of droplets to accumulate in the corners. But one disadvantage in solving DFP is, if droplets take their initial velocity from DFP, their traveling direction might deviate somewhat from the true solution. Further comparison among these solvers is discussed in the next chapter.

4.2 PlicIf solver

4.2.1 Boundary conditions for PlicIf

The boundary conditions for PlicIf include data for element *vof* and specific enthalpy, which are applied on all the faces around the PlicDomain. The element *vof* and specific enthalpy are mapped from the boundary elements to the boundary nodes using the utility *Element2NodalData*. Boundary conditions are defined by boundary nodes and two boundary nodal data arrays are input to the PlicIf solver.

4.2.2 PlicIf domain

There are two ways to make a PlicIf domain. One way is to just use the filling domain. This is easy to code, but we have to solve PlicIf on all elements, including the elements whose *vof* value is zero during one step. The second way to make a PlicIf domain is to grow several layers (say, specify 20 layers in design file) from non-zero *vof* elements into the set of zero *vof* elements. Then make the PlicIf domain as the union of the elements in these 20 layers with the non-zero *vof* domain. An element set named *PlicHaltSet* is also introduced to halt the PlicIf solver as soon there is liquid flowing out of the PlicIf domain.

4.2.3 Ramp up velocity

In practice, there are both a fast and a slow shotSleeve velocity. The control from slow shot velocity to fast shot velocity is set as follows: after each PlicIf solving, check whether there is some liquid in the casting or not. If yes, solve ERK once to ramp the inlet velocity from low to high. In practice, slow shot is used to move liquid from the shot sleeve to fill the runner with minimum turbulence and trapped air.

4.2.4 Determine the PlicIf time interval

The time step for PlicIf is determined from the Courant number and the maximum velocity after ERK solving. Based on the Courant number, the bigger the maximum velocity, the smaller the time step that will be used for PlicIf, and the smaller the amount of liquid advected in each PlicIf step. Sometimes, in order to guarantee a normal filling rate, the max velocity from ERK solving is limited. This could be realized by setting an upper velocity limit to the ERK solver.

4.3 Air exhaust

In the filling analysis, the air exhausts are treated as Neumann boundary conditions. Liquid flows out of the domain through air exhausts. There are two ways to deal with this part. One is using a pressure boundary condition. The other way is using airSinkLeak. These two models are discussed below.

4.3.1 Pressure boundary conditions

For the model of pressure boundary condition, we introduce three element-set lists as follows:

1. *outletElSet*: The length of this list does not change during the filling process.

It is obtained by growing three layers from each airExhaust into the overflow

and obtaining a corresponding layer set, making a list of all these layer sets and name it *outletElSet*. For example, if there are eight airExhausts in the STL file, we should get a list composed of eight sets.

2. Make a copy of the list *outletElSet*, and give it a new name, *pressurBCOutletSet*.

The difference for list *pressurBCOutletSet* from *outletElSet* is that its length will keep changing during the filling process.

3. If we continue growing one layer from *outletElSet* into overFlow/casting. The result list by assembling obtained layer sets is named *pressureBCHaltSet*.

First, obtain the volume to fill in each element set in *pressurBCOutletSet* list. This is the volume that we want to fill completely before changing this boundary condition from pressure to zeroDirichlet. After each PlicIf step, some liquid might enter the elements in *pressureBCHaltSet*. Find and accumulate the amount of liquid, then clear it from *ERKDict*. Keep collecting and clearing liquid until the total accumulated liquid is equal to the volume to be filled in *pressurBCOutletSet*. Then delete this pressure B.C. from *pressurBCOutletSetlist*, and add the elements to DirichletBCSet, i.e., change the pressure B.C. to Dirichlet B.C.

In the pressure-boundary model, filling domain is made by differencing the *outletElSet* from the union of runnerGate, casting and overflow sets.

4.3.2 Cavity air

The second way to model air exhausts is through making *airSinkLeak*. An ERK solver needs two *airSinkLeak* element sets as arguments. One is at time t_1 , the other is at time t_3 . We create the same sets *airSinkLeaks_{t1}* and *airSinkLeaks_{t3}* by growing several layers from the *airExhaust* sets at the beginning. When running ERK, the solver will update the *airSinkLeaks_{t3}* based on the vof field. After ERK solving, update *airSinkLeak_{t1}* with *airSinkLeak_{t3}*, and empty *airSinkLeak_{t3}*. With the progress of filling, the volume of both *airSinkLeaks* will become smaller and smaller. The solver halts when the size of *airSinkLeak_{t3}* after an ERK solve is zero.

4.4 Cavity air and trapped air

During the PlicIf filling process, many air bubbles are generated in the domain. These bubbles are in a dynamic state, continuously splitting and merging with each other. The codes below are used to compute the state parameters for each air bubble for each time step. The state parameters for a bubble include *pressure*, *temperature*, *volume* and *mass*.

4.4.1 Initialize cavity air

At the beginning, the cavity air domain is the whole filling domain. The value of initial pressure and temperature is read from the design file and applied to all the elements in the cavity air domain.

4.4.2 Find separated trapped air

Loop over all the elements in the domain to find elements whose vof is not one, i.e., if there is some air in the cell. The union of such elements is named $VofE01E0$, which includes both $VofE0$ set (air only) and $VofE01$ set (air and liquid). The volume of air and liquid in $VofE01E0$ is also obtained respectively for use later. Then find all connected subsets in $VofE01E0$. These subsets are candidates for updating cavity air.

4.4.3 Get relations and merge trapped air

For a specific time step, an air cavity or bubble is generated either from splitting or merging parent air cavities. In order to obtain proper initial value for state parameters of an air bubble, we need a relationship between parent air bubbles and child air bubbles first. In the code, create a big list - *parentChildList* (for each parent, list its children). There are different families in this list. In one family, the first element is the parent air bubble (only one), the rest are the child air bubbles (these may be

quite a few). In a similar way, we create a *childParentList* (for each child, list its partents).

A case that needs special attention is that one child has more than one parent air bubbles. In this case, we need to map data from all parents in order to determine the proper initial state for this child. To do this, we merge the parents who share one child together to make a bigger family.

4.4.4 Update cavity air

First read state data for a parent cavity air, including pressure, temperature and volume. Based on the ideal gas law, calculate the number of moles in the parent cavity air set:

$$MolesParent = P \cdot V / R \cdot T$$

where, P is pressure, V is volume, T is temperature and R is the gas constant. The mass of parent cavity air could be obtained from its mole numbers:

$$MassParent = MolesParent \times MoleWeight$$

Find the number of elements in each child air cavity. Compute the ratio of the number of elements in this child to the number of elements in all children of this parent. Use this ratio to distribute the mass of parent to each child:

$$MassChild = \frac{nbElementsThisChild}{nbElementsAllChildren} \times MassParent$$

Two stages are assumed during the state change from parent to child air cavity. In the

first stage, the *MassChild* is changed due to leaks, but the pressure is kept constant.

While in the second stage, the mass is kept constant, and the pressure changes.

The mass computed above, *MassChild*, is before any state change. During the PlicIf filling process, some liquid might flow out of the air exhaust. The mass leak rate could be computed based on the parameters of a specific air exhaust. After a certain time interval, the leaked mass is subtracted from *MassChild* to obtain the mass at time t_1 :

$$MassChild_{t_1} = MassChild - TotalMassLeakRate \times dTimePlic$$

Compute volume after leak as:

$$Moles_{t_1} = MassChild_{t_1} / MolWeight$$

$$Volume_{t_1} = Moles_{t_1} \times R \times TempParent / PressureParent$$

where R is the universal constant, taken as $8.3149 \times 10^3 \text{ Jole/kg} \cdot \text{mole} \cdot \text{degree}$.

In the second stage, start from the changed volume to find the pressure after the change.

$$Pressure_{t_2} = Pressure_{t_1} \times \left(\frac{Volume_{t_1}}{Volume_{t_2}} \right)^\gamma$$

where, $Volume_{t_2}$ is computed from the number of elements in this child air cavity:

$$Volume_{t_2} = nbElemThisChild \times dx \times dy \times dz$$

And $Pressure_{t_1}$ is the same as $PressureParent$ due to the constant pressure in the first stage. As can be seen from the above manipulation, the stage of the air cavity at time t_1 is just assumed.

Applying the ideal gas law,

$$Temperature_{t2} = Pressure_{t2} \times Volume_{t2} / (Moles_{t2} \times R)$$

where $Moles_{t2}$ equals to $Moles_{t1}$ due to the constant mass in the second stage.

At this point, the state of the air cavity has been updated from parent state to child state, including pressure, temperature and mass.

4.4.5 Update parent list

After updating the state for child air cavities, upgrade these children to new parents and delete the old parent air cavities. The new parent air cavity list is kept for use in the next time step.

4.4.6 Trapped air cavities

Trapped air cavities are those air cavities that are not connected to any air exhausts. After updating all air cavity states, trapped air is filled based on the pressure ratio, i.e., the volume change of trapped air.

Find pressure ratio

Before updating a child air cavity, the maximum pressure in each parent air cavity is found and named P_1 . Similarly, after updating, seek P_2 , the maximum pressure in each new parent air cavity. Usually, we expect the pressure ratio, P_1/P_2 , should be

less than one, otherwise the air cavity is expanding.

Volume of liquid to fill due to pressure change

Find all the air volume in the filling domain. Subtracting legal air cavities (those connected to air exhausts) from it gives the trapped air. The change of air volume due to the change of pressure is obtained as follows:

$$volumeOfTrappedAirChange = volumeOfTrappedAir \times (1.0 - pressureRatio^{\frac{1}{1.4}})$$

This is the volume of liquid to distribute to trapped air.

Fill trapped air and map enthalpy

To distribute the liquid to trapped air, start from *voeE01* set. There is both liquid and air in the elements of this set, i.e., $1.0 > vof > 0.0$. Compare the volume of air, V_{air} , in the elements of this set with the volume of liquid, V_D , to distribute. If V_D is larger, set value of *voe* to be 1.0 in all elements in the *voeE01* set. Then $V_{D2} = V_D - V_{air}$ is the remaining liquid to distribute. Make a new *voeE1* set by union old *voeE1* set with filled *voeE01* set. Then grow one layer from new *voeE1* set into trapped air element sets. Compare the available volume of air in the new layer set with remaining liquid, V_{D2} . Keep growing layers until all of the liquid to distribute is distributed completely to trapped air elements.

Figure 4.1: Data mapping from donor to acceptor elements

When distributing liquid to trapped air, the proper enthalpy value of newly-filled elements must be specified. As shown in Figure 4.1, there are donor elements and acceptor elements. The basic step is mapping enthalpy from donor elements to their corresponding acceptor elements. Find the boundary face nodes of the donor element set and the acceptor element set separately. Make the intersection set of these two face nodal sets. For each face node in the intersection, find its two sharing elements. One element is in the donor set, while the other is in the acceptor set. Next, map enthalpy from donor elements to their corresponding acceptor elements. One special case is shown in element (A). It does not have a neighboring donor element. Here, it is necessary to have a second mapping, i.e., mapping enthalpy from new donor

element (B) and (C) to this special corner element (A) in Figure 4.1. Another special case is element (D). It has two donors. The proper enthalpy value for element (D) should be mapped from both (F) and (E).

Fill all air cavities at the end

When the size of the total air exhausts becomes zero, halt the filling solver and fill the rest of the volume by just growing layers. This is the same idea as filling trapped air.

4.5 Restart the job

For a practical job, the filling analysis may take several days to run. During this process, the job might need to be restarted. For example, if the filling process is very slow after it fills 40%, we may want to halt the job to change some parameters to make it run faster. In this case, it could save a lot of time to keep the available good results and continue the running from the finished 40% instead from the very beginning. There is a restart capability in the filling solver to realize this function.

For a filling job, two sets of data are kept in *ERKDict*. One set is for restart, the other set is for visualization and postprocessing. All the data fields are necessary for the restart purpose, while we only keep certain data fields for postprocessing. More visualization result files are kept than restart result files in order to get high resolution

visualization with respect to time.

In order to do a restart, load the data from restart result file into *ERKDict* based on a specified time step number.

For pressure B.C., we create fields in *ERKDict* and save the state of boundary conditions to restart result files. When doing restart, the pressure BC is read from *ERKDict* as the other data.

4.6 How to run a filling analysis

Generally, the steps in running a filling job are creating STL files, creating design file, creating geometry file, running the filling analysis and postprocessing the results.

4.6.1 STL files

Usually, STL files are exported from solid model files on CAD system. STL files are composed of triangles and vertices, with different files to represent different parts. A careful check of STL files is necessary before meshing to make sure different parts are connected properly. No overlap should exist between different STL parts.

4.6.2 Design file

All necessary input data for an analysis is prepared in the design file. A design file is composed of different sections. Each section starts with “@<section title>” and ends with “@ENDOFSECTION”. Usually, the following information are specified in a design file:

Part information

Including *castingPartsInformation*, *overflowPartsInformation*, *runnerGatePartsInformation*, *shotSleeveInformation*, *diePartsInformation* and *airExhaust*.

In each part-information section, we specify the part name, STL file name and number. For example:

```
@runnerGatePartsInformation
nameOfStlFile$    runner1.stl    runner2.stl
numberOfStlFiles%    1    1
runnerGatePartName$    Runner1    Runner2
@ENDOFSECTION
```

In *shotSleeveInformation* section, we also specify the following inlet data:

1. velocity: slow shot velocity and fast shot velocity.

2. unit velocity vector: for example, if liquid is poured in negative y direction, we have unit velocity vector as $0 \quad -1 \quad 0$.
3. time list for slow shot: generally has five real numbers.
0.0: start time; 0.003: first ramping end time; 200.0: second ramping start time; 200.003: second ramping end time; 300.0: end time.
4. time list for fast shot: at the beginning, we have exactly the same numbers here as slow shot time list. When the liquid reaches the casting, i.e., at the time to ramp up inlet velocity from low shot to fast shot, the filling solver will change the third number - second ramping start time to the global time at that moment. Also the fourth number will be changed correspondingly.
5. slowFastShotSwitch: at the beginning of the filling analysis, this flag number is 1. After shot velocity is ramped up to fast shot velocity, the solver will change the flag value to 0.
6. switchResNb: the step number to ramp is recorded here.

Special attention should be paid to the values in 4, 5, 6 when doing a restart near the ramping-up point.

Filling solver parameters

Filling solver parameters are specified in two sections, *FillingSolverOptions* and *fillingSolverParameters*:

- `copyResAtPercentFilled`: controls the frequency of copying ERKDict to visual result files.
- `filledPercentForHaltPlicIf`: control when to halt PlicIf solver in one step.
- `PlicStepsE01ToE1`: changing all vofE01 elements to VofE1 after how many PlicIf steps.
- `PLicStepToDecideCopy`: checking percent filled to decide when to copy ERKDict to result files after how many PlicIf steps.
- `IFGravityDirection`, `GravityDirec1`, `GravityDirec2`, `GravityDirec3`: specify gravity direction vector.
- `nbPlicDomainLayers`: how many layers to grow from VofE1 set to make the PlicDomain.
- `nbAirSinkLayers`: how many layers to grow from each air exhaust to make air sink leak sets.
- `criterionForDivCheck`: criterion to do div free check, if divergence of any cell is larger than this value, the cell would be printed out as a bad element.

- `castingMaterial`: the specified material name must come from Mrco material library.
- `iniTimeStepSize`: initial value for ERK time step size. ERK solver could choose its own time step size after solving, but does require an initial guess.
- `ERKuHatTolerance`: criterion to solve for $\hat{u}$ in time-stepping of PDE.
- `DFPTolerance`: if the maximum divergence of an element exceeds this value, then solve DFP to make velocity field div free.
- `lowCN`, `highCN`: limits on Courant number for solving ERK. If the Courant number after ERK is not in this range, ERK solver chooses a new time step and resolve this time step.
- `maxVelocityLimit`: velocity upper limit in solving ERK, the solver will reduce any velocity that exceeds this limit to this limit.
- `maxIfTimeSteps`: the max number of PlicIf steps that could be solved in one loop.
- `maximumPlicCN`: courant number for PlicIf solving, usually 0.95. It must be ≤ 1.0 .

Air exhaust parameters

In *AirExhaustParameters* section, data are given for each air exhaust. The data include: height, width, length, *surfaceRoughness*, and *frictionFactor*. Mass leak rate could be calculated for an air exhaust based on these parameters.

Cavity air parameters

In *airParameters* section. The values are specified for:

- *ambientTemperature*: ambient atmosphere temperature outside die (in Kelvin).
- *initialTemperatureAirInTheCavity*: initial cavity air temperature.

In *InitialCavityAirParameters* section, we have:

- *PressureInitial*: value to initialize cavity air pressure.
- *TemperatureInitial*: value to initialize cavity air temperature.

Other parameters

There are other sections in a design files, such as:

- *meshData*: data used for making a grid file.
- *spotCoolingLineModel*, *coolingLineGeneralInformation*, *flowThroughCoolingLineModel*: data for cooling line analysis.

- oilLineModel: data for oil line analysis.

4.6.3 Meshing - make grid file

When STL files and design file are available, we need to mesh the geometry for analysis, i.e., create grid file. The elements chosen are 6-node or 8-node brick elements. A 6-node brick element is used for ERK, DFP, PlicIf and DDAE solvers. A 8-node brick element is used for the Droplet solver. The size of elements in the grid file to mesh depends on how accurate result is expected. The number of elements in the result mesh obtained from the mesher utility is usually between 90% and 110% of the size we specified. The mesh generated in this way is a structured mesh. Often all elements are cubes and all have exactly the same size.

4.6.4 Different filling solvers

Usually, in a filling analysis, Navier-Stokes equation is solved for the velocity and pressure, then free surface equation / Droplet equation are solved to get proper *vof* distribution in the filling domain, i.e., the liquid advection pattern. Currently, Mrco group has developed four types of filling solvers:

1. (ERK solver) + (PlicIf solver).
2. (ERK solver) + (DDAE solver).

3. (Droplet solver) + (DFP solver).

4. (Droplet solver) only.

4.6.5 Boundary conditions and initial conditions

In order for the finite element analysis to solve the partial differential Navier-Stokes equation, a set of boundary conditions must be specified. Boundary conditions must be specified for every boundary face. On the inlet, the velocities are specified at the shot sleeve. In a practical die casting process, the designer has control over the velocity of the plunger. To do different analysis with different inlet flow rates, different boundary conditions at the shot sleeve must be specified. A zero velocity BC is specified on faces adjacent to the die. At the outlet, a pressure BC is specified on faces adjacent to an air exhaust. The pressure value could be set to be atmosphere pressure or zero pressure depending on the specific situation in practice.

4.6.6 Material properties

The name of casting material used should come from Mrco material library. For example, if aluminum alloy *Al380* is specified. Its density read from Mrco library is $\rho = 2390 \text{ kg/m}^3$. The dynamic viscosity is $0.0012 \text{ N} \cdot \text{s/m}^2$.

Chapter 5

Numerical Results

In first half of this chapter, different solvers are tested on an automobile transmission filling analysis. *vof* results are postprocessed and different flow patterns of metal liquid are compared. The numerical results presented below are from:

1. PlicIf solver with changing pressure BC (pressure $\rightarrow$ zeroDirichlet).
2. Droplet solver.
3. Droplet solver with DFP solver.

In the second half of this chapter, a comparison between some experimental specimens and numerical simulation results is made to show the effectiveness of the filling solver prediction.

5.1 Transmission test

Figure 5.1 is the STL files for an automobile transmission part that is made by die casting. About 20 kg of liquid Al-11 % Si alloy is injected into a H13 steel die in about 30 milliseconds. The liquid solidifies and cools sufficiently in about 30 seconds to be ejected from the die. It requires about 30 seconds to open the die, eject the castings, clean the die and prepare for the next shot. This is roughly a 60 second cycle. The die is cooled by water or oil lines.

The filling process is simulated by different solvers with the same boundary and initial conditions. The transmission is meshed into a 150K element grid file. The low shot sleeve velocity is set to be 0.69 m/s, the fast shot sleeve velocity is set to be 2.89 m/s. Depending on the type of solver, the number of result steps is different. Nine steps for each solver are chosen to show the basic flow pattern.

5.1.1 PlicIf with changing pressure B.C.

The transmission is filled completely after 51 steps. From the brief flow pattern, Figures 5.2 & 5.3, it shows the casting half near the inlet is filled faster than the half away from it.

5.1.2 Droplet solver

A total of 57 result files are obtained after solving. From *vof* advection in Figures 5.4 & 5.5, the front face of metal liquid in the casting is almost straight. This could be understood as follows: when a structured mesh is used as bouncing layer, considering the geometry of transmission inlet part is complex, most inlet droplets would not bounce properly through runner gate into the casting. As a result, *vof* accumulates at the inlet. The way to redistribute the extra *vof* by growing layers could make the advection of metal liquid looks diffusive. As mentioned earlier, two major improvements to this solver would be using umesh or STL files for computing normals of bouncing layer elements and adding Newmark method. Umesh bouncing layer could help droplets travel by bouncing through runnergate easier. While Newmark method will guarantee the incompressibility condition of the metal liquid, i.e., no need to redistribute extra *vof*.

5.1.3 Droplet solver with DFP solving

In Figures 5.6 & 5.7, the simulation results from Droplet DFP solver is shown. As mentioned earlier, solving DFP after each step of droplet advection would change the velocity direction for those piled up droplets, decrease the diffusivity brought by redistributing extra *vof*. It is clearly shown that the flow situation in Droplet DFP solver looks more realistic than Droplet solver. Comparatively, the filling pattern

from Droplet DFP solver is similar to that from PlicIf solver, but the former takes only half of the running time of the later in solving the job. Moreover, the filling rate of Droplet DFP solver could be controlled much easier than PlicIf solver.

5.2 Bearing plate test

The part in Figure 5.8 is a bearing plate used as an automobile part. Special experiments were made to show the flow pattern of metal liquid during the filling process. PlicIf solver with changing pressure BC and Droplet DFP solver were tested on this job (100K elements). Numerical results were postprocessed and compared with experiment results. The low shot sleeve velocity was taken as 0.635 m/s, while the fast shot sleeve velocity was taken as 2.43 m/s.

5.2.1 Practical experiment

In the experiments, different amount of metal liquid was poured into the shotsleeve of the same die. After each shot, an incomplete casting part was obtained. The completeness of each part depends on how much liquid was poured into the mould, which could be reasonably assumed to represent different filling stages. Both top and bottom views for incomplete parts are shown in Figures 5.9 & 5.10. Also, a complete part was cut into pieces to check for porosity.

5.2.2 Numerical results from the solvers

Compared with the shape of incomplete experiment parts, the closest filling steps are picked out from the numerical results of both PlicIf solver and Droplet DFP solver, Figures 5.11 to 5.18. It is shown that, on the first and second stages, the simulation results from both solvers match well with the shape of the experiment specimens, while the Droplet DFP solver seems slightly more accurate. In the last two stages, some difference between the experimental results and numerical prediction arises on the last-filled regions in the casting. Close examination reveals that the results from Droplet DFP solver match a little bit better with the shape of specimen than PlicIf solver.

5.2.3 Prediction of trapped air

In Figure 5-19, some pieces cut from a complete bearing plate part are shown. It is shown that some air bubbles are trapped in the downstream of the ring. This problem is predicted clearly. Please refer to Steps 24, 25 from numerical results of PlicIf solver.

Figure 5.1: Different Perspectives of A Transmission (STL files)

step 7

step 15

step 20

step 25

step 30

step 35

step 40

step 45

step 48

Figure 5.2: PlicIf solver with Changing Pressure Boundary Conditions I

step 10

step 15

step 20

step 25

step 30

step 35

step 40

step 45

step 48

Figure 5.3: PlicIf solver with Changing Pressure Boundary Conditions II

step 10

step 15

step 20

step 25

step 30

step 35

step 45

step 50

step 55

Figure 5.4: Droplet Solver I

step 10

step 15

step 20

step 25

step 30

step 35

step 45

step 50

step 55

Figure 5.5: Droplet Solver II

step 5

step 10

step 15

step 20

step 25

step 30

step 40

step 50

step 60

Figure 5.6: Droplet DFP solver I

step 5

step 10

step 15

step 20

step 25

step 30

step 40

step 50

step 60

Figure 5.7: Droplet DFP solver II

Figure 5.8: Different Perspectives of Experiment Specimen

Figure 5.9: Front View of Experiment Specimen

Figure 5.10: Back View of Experiment Specimen

Figure 5.11: Comparison I – PlicIf Solver

Figure 5.12: Comparison II – PlicIf Solver

Figure 5.13: Comparison III – PlicIf Solver

Figure 5.14: Comparison IV – PlicIf Solver

Figure 5.15: Comparison I – Droplet DFP Solver

Figure 5.16: Comparison II – Droplet DFP Solver

Figure 5.17: Comparison III – Droplet DFP Solver

Figure 5.18: Comparison IV – Droplet DFP Solver

Step 24

Step 25

Figure 5.19: Prediction of Trapped Air in Casting

Chapter 6

Conclusion and Future Work

6.1 Conclusions

The algorithm and numerical results of PlicIf and Droplet filling solvers are presented and compared in this thesis. The numerical results from PlicIf solver basically match well with the actual flow of metal liquid in simple geometry castings. The problem with the PlicIf filling solver is that the liquid front face tends to follow the momentum of the air. As an improvement of PlicIf solver, a droplet solver was developed. Using a different algorithm, the metal liquid in this new solver is treated as numerous droplets. Droplets are initialized with certain diameter and position either at the inlet of filling domain (inlet droplet) or inside the filling domain (vof droplet). This is a Lagrangian solver and droplets travel in the die casting starting with initial velocity. The drag

force from air and bouncing between die walls control the trajectory of each droplet. After travelling for a certain period of time, the droplet solver is halted and the lumped liquid points along the travel path of inlet droplet will be mapped into the elements in which they fall. For a vof droplet, the only lumped liquid point is at the end of the travel path. After the mass and momentum is mapped from droplets to elements, DFP could be solved to get a better velocity field to help the droplets travel around corners. The surface layer elements on the inner side of the die wall are treated as bouncing elements. For droplets to bounce properly, the normals for the bouncing elements are calculated. As an improvement of the droplet solver, the normal calculation could be obtained from the STL file or an unstructured mesh instead of from a structured mesh. This helps to decrease the piling up of droplets at the die casting inlet. However, even with DFP solving, the piling of droplets at corners could not be avoided completely. As a result, after each step of filling, we are trying to redistribute the extra vof to their neighbouring elements. The result of redistributing the extra vof makes the metal flow pattern diffusive.

6.2 Future work

In the new solver, we expect to fix the problems existed in the current filling solvers. The major improvements include:

- Using STL files to compute the normals for bouncing elements.
- Consider the drag force exerted on the liquid droplets by the air. If this drag force is too high. The moving of droplets are going to follow the air momentum exactly as the case in PlicIf solver. If this drag force is too low, the liquid droplets are easily piled up in complex geometry areas, as we met in the first version of droplet solver. Considering the drag force could be affected by varying the air pressure and air viscosity, further numerical tests are conducted to determine the proper value of drag force.
- Using a more efficient way to distribute the extra vof. Stress solver will be used to change the droplet positions.
- Keep only one type of droplets, this could conserve the droplet mass and momentum exactly and improve analysis accuracy.
- May consider the interactions between droplets to give a better numerical results.

Bibliography

- [1] Trevor Sawchyn, *Die Casting Design*, course project, Carleton University. 1999.
- [2] E.A. Herman, *Die Casting Dies, Designing*, Society of Die Casting Engineers. 1985.
- [3] Marco Guerra, *Die Casting Design - A Parametric Approach*, Department of Mechanical and Aerospace Engineering, Ottawa-Carleton Institute for Mechanical and Aerospace Engineering. 1997.
- [4] D.L. Youngs, *An Interface Tracking Method for 3D Eulerian Hydrodynamics Code*, Atomic Weapons Research Establishment, Aldermaston, Reading, RG7 4PR, U.K., 1984.
- [5] Liang Zhou, *Modeling Interface Advection in the Die Casting Processes*, Master thesis, Carleton University, 1998.

- [6] Mehmet Naci Dai, *Automating the Analysis of Complex Physical System - The Virtual Foundry*, Ph.D thesis, Carleton University, 1994.
- [7] S.A. Sjolander, *Fundamentals of Fluid Dynamics*, Course Notes, Carleton University, 1998.
- [8] Edward Hughes, *Computation of Time-Dependent Incompressible Navier-Stokes flows*, Carleton University, paper, 2000.
- [9] D.D. Klada and D.P. Georgiou, *A Relative Examination of $C_p - Re$ Relationships Used in Particle Trajectory Calculations*, Journal of Fluids Engineering, Vol. 115, March, 1993.
- [10] Darin Diachin, Lori Freitag, Daniel Heath, *Interactive Simulation and Visualization of Massless, Massed, and Evaporating Particles*, Argonne National Laboratory. 1998.
- [11] Naci M. Dai, John A. Goldak, *Computational Models for Cold Shuts*, paper, Carleton University. 1994.
- [12] H.H. Pokorny, P. Thukkaram, *Gating, Die Casting Dies*, The Society of Die Casting engineers, 1981.
- [13] Sulsky D, Zhou SJ, Schreyer HL, *Application of a Particle-In-Cell Method to Solid Mechanics*, Computer Physics Communications, Vol.87 No.1-2. 1995.